An Introduction to

`(re.ex|re+gex|re?gex|re*gex){1}`

William Smith

Professional Services Enginerd, Jamf

http://entourage.mvps.org/

http://entourage.mvps.org/blog

~~http://entourage.mvps.org/blog~~

$$\begin{tikzcd}\text{http://entourage.mvps.org/blog} \arrow[d] \\ \text{http://officeformachelp.com}\end{tikzcd}$$

entourage

http://entourage.mvps.org/blog

$\downarrow$

http://officeformachelp.com

.htaccess

http --> https
talkingmoose.net --> www.talkingmoose.net

.htaccess

```
RewriteEngine on
RewriteCond %{HTTP_HOST} ^entourage\.mvps\.org/blog$ [NC]
RewriteRule (.*)$ http://officeformachelp.com/$1 [L,R=301,NC]
```

.htaccess

```
RewriteEngine on
RewriteCond %{HTTP_HOST} ^entourage\.mvps\.org/blog$ [NC]
RewriteRule (.*)$ http://officeformachelp.com/$1 [L,R=301,NC]
```

Agenda

What is regex?

Characters with special meanings

Character classes and grouping

Applications and command line tools that support regex

Examples from real world experiences

Regex resources

What is regex?

Short for "regular expression"
"Regular" comes from the concept of a "regular language"

alphabet = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, - }
words = { 02134, 02134-3611, 55119, 55119-5027, 90210, 90210-0802 }
language = United State Postal Service zip codes

A regular language contains a finite number of words.
We can use an algorithm to determine whether a word belongs to a language.

What is regex?

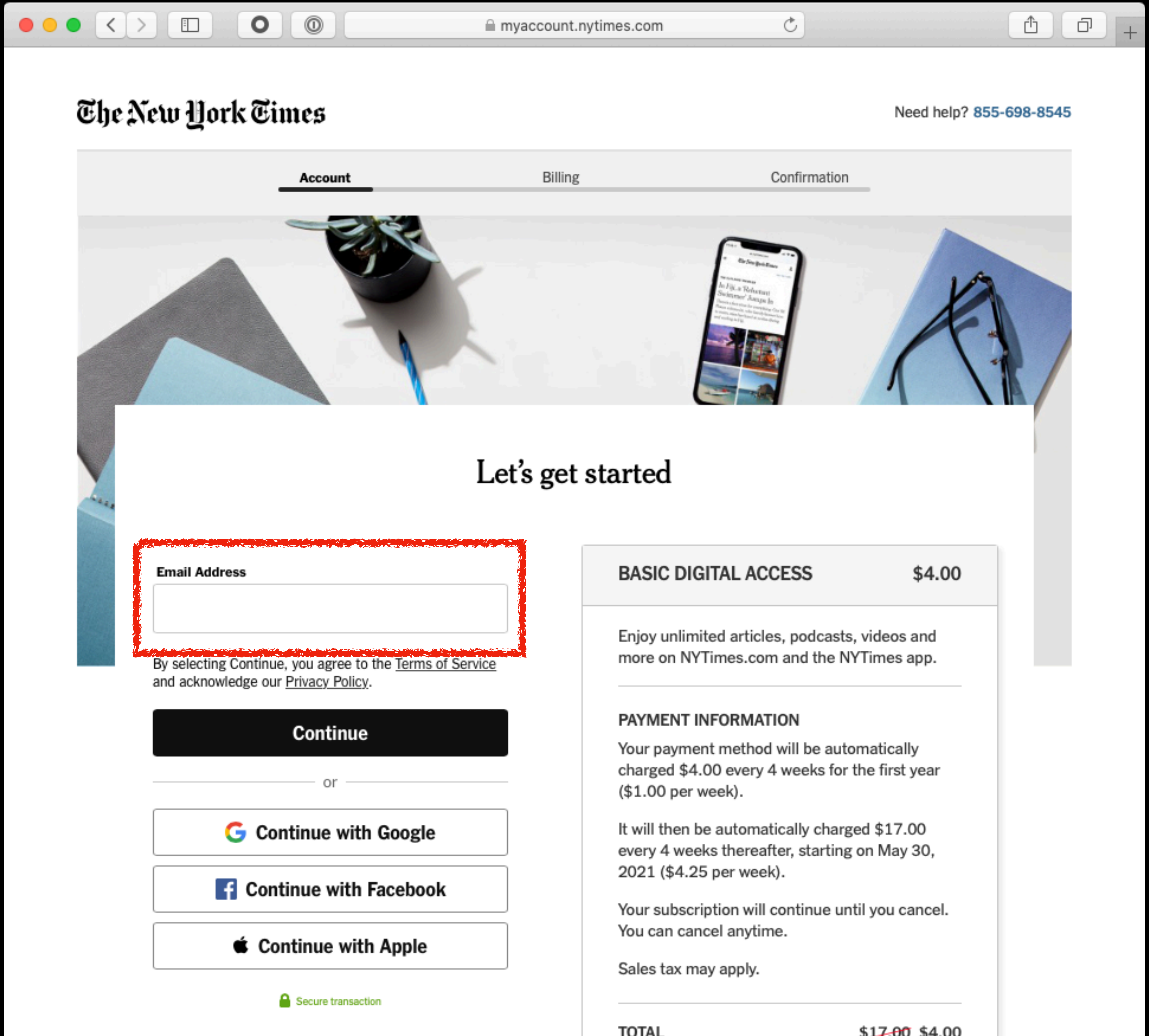
United States Postal Service zip code algorithm = ##### or #####-####

regular expressions (regexes, regexp or regexen) = patterns

Regex is pattern matching.

Regex is pattern matching.

Validation.



myaccount.nytimes.com

The New York Times

Need help? 855-698-8545

Account Billing Confirmation

Let's get started

Email Address

By selecting Continue, you agree to the Terms of Service and acknowledge our Privacy Policy.

Continue

or

Continue with Google

Continue with Facebook

Continue with Apple

Secure transaction

BASIC DIGITAL ACCESS $4.00

Enjoy unlimited articles, podcasts, videos and more on NYTimes.com and the NYTimes app.

PAYMENT INFORMATION

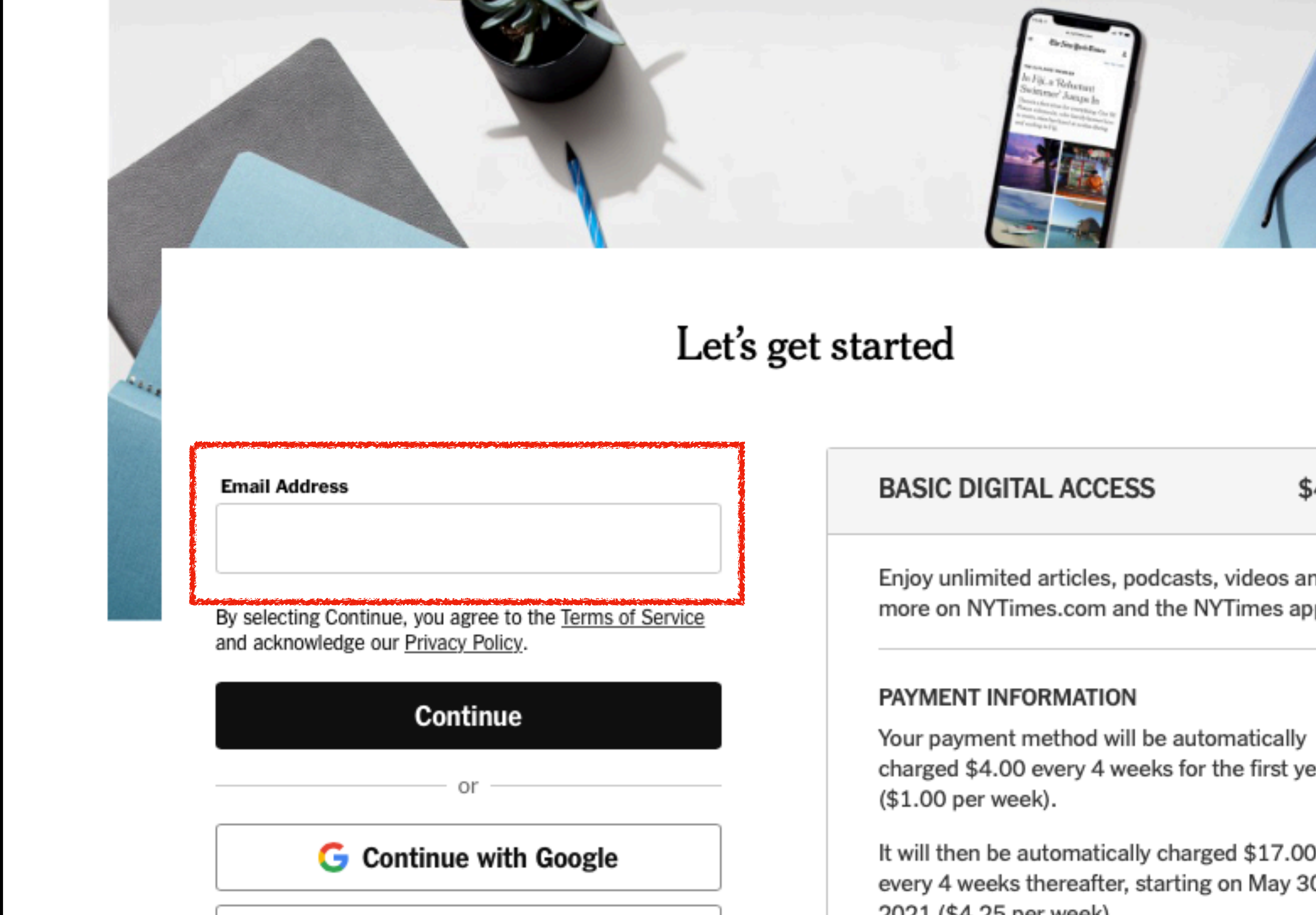
Your payment method will be automatically charged $4.00 every 4 weeks for the first year ($1.00 per week).

It will then be automatically charged $17.00 every 4 weeks thereafter, starting on May 30, 2021 ($4.25 per week).

Your subscription will continue until you cancel. You can cancel anytime.

Sales tax may apply.

TOTAL ~~$17.00~~ $4.00



Let’s get started

Email Address

By selecting Continue, you agree to the Terms of Service and acknowledge our Privacy Policy.

Continue

or

Continue with Google

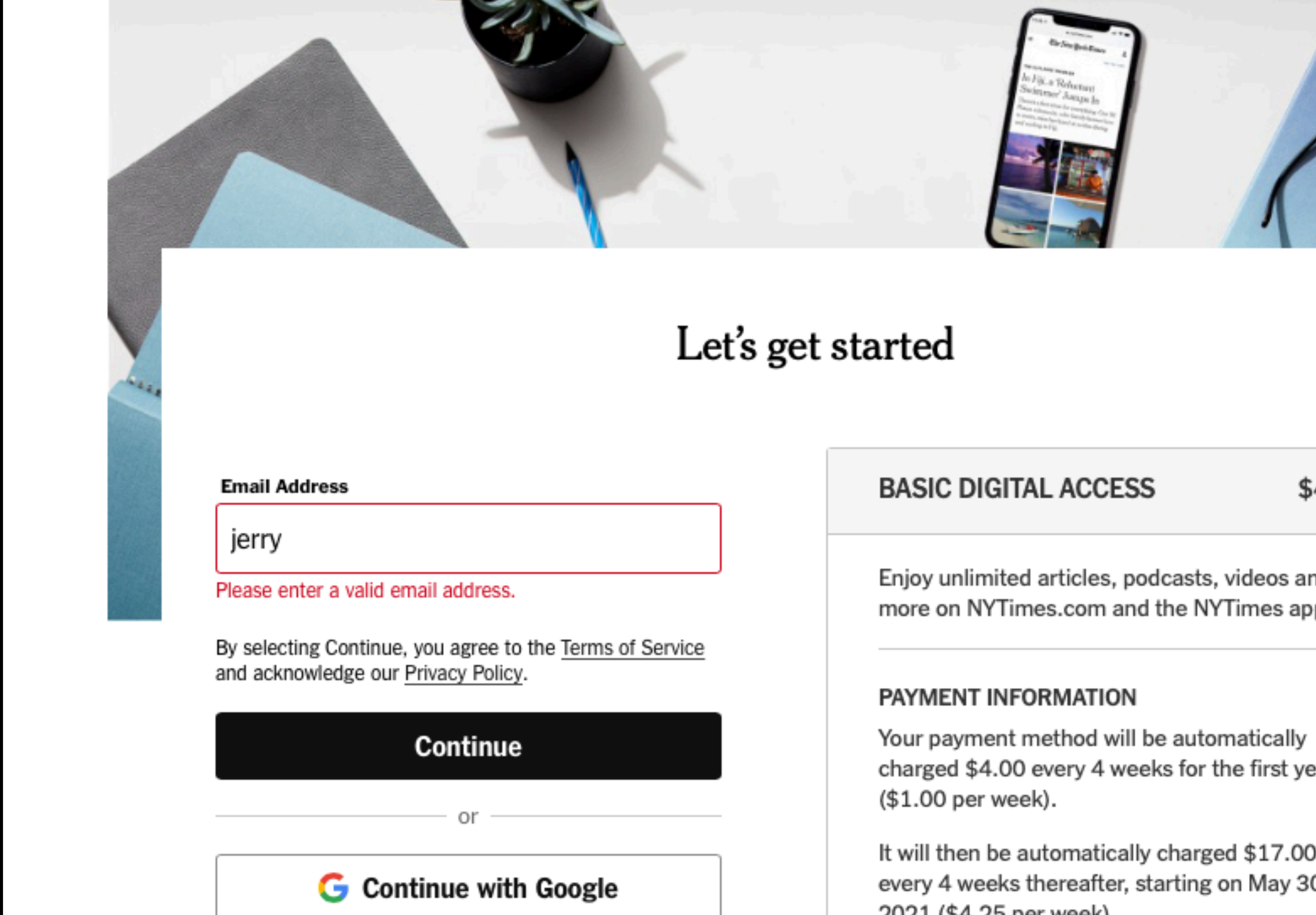
BASIC DIGITAL ACCESS $

Enjoy unlimited articles, podcasts, videos an
more on NYTimes.com and the NYTimes ap

PAYMENT INFORMATION

Your payment method will be automatically
charged $4.00 every 4 weeks for the first ye
($1.00 per week).

It will then be automatically charged $17.00
every 4 weeks thereafter, starting on May 3
2021 ($4.25 per week).



Let’s get started

Email Address

jerry

Please enter a valid email address.

By selecting Continue, you agree to the Terms of Service and acknowledge our Privacy Policy.

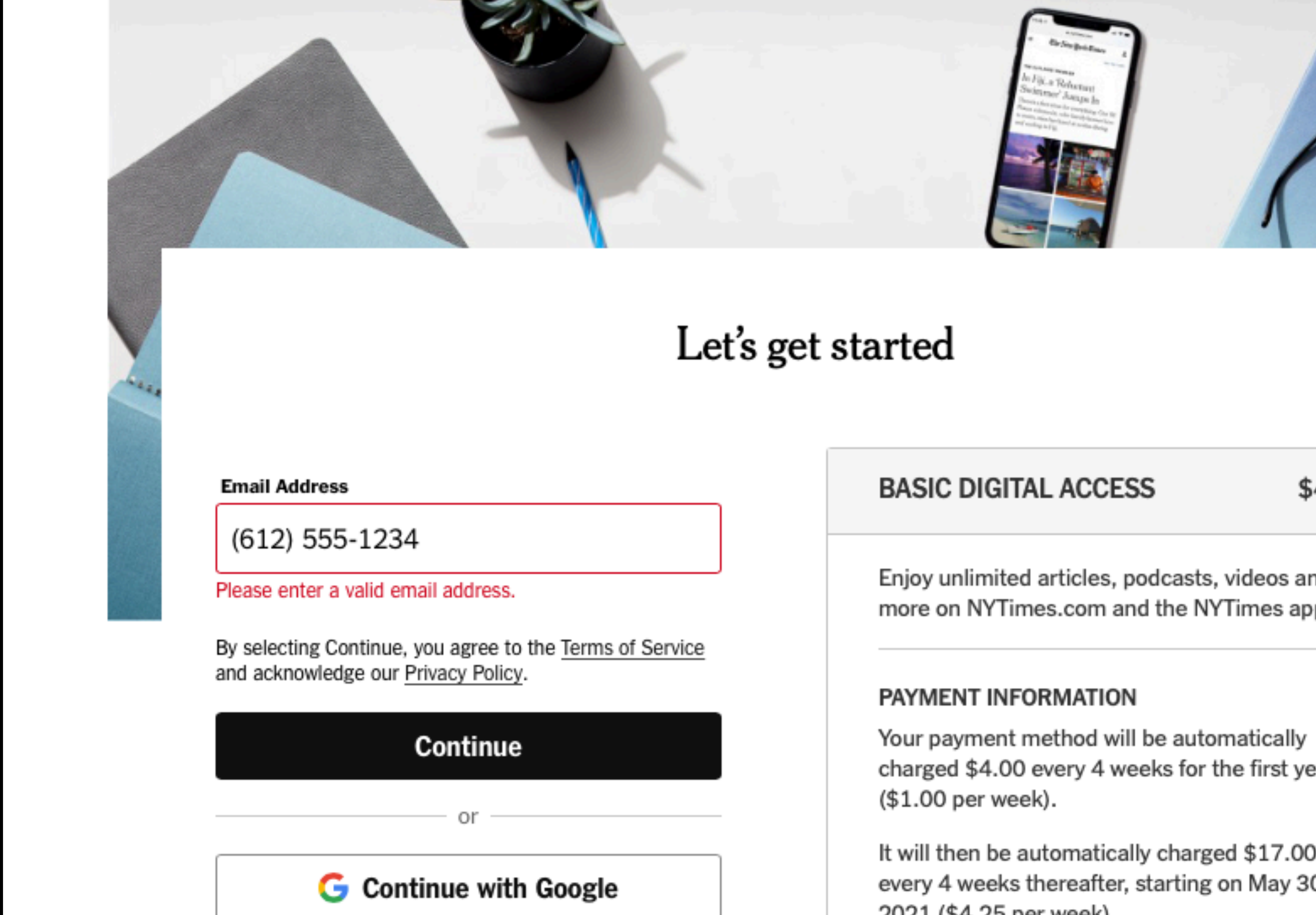
BASIC DIGITAL ACCESS $

Enjoy unlimited articles, podcasts, videos an
more on NYTimes.com and the NYTimes ap

PAYMENT INFORMATION

Your payment method will be automatically
charged $4.00 every 4 weeks for the first ye
($1.00 per week).

It will then be automatically charged $17.00
every 4 weeks thereafter, starting on May 30
2021 ($4.25 per week)



Let’s get started

Email Address

(612) 555-1234

Please enter a valid email address.

By selecting Continue, you agree to the Terms of Service and acknowledge our Privacy Policy.

Continue

or

Continue with Google

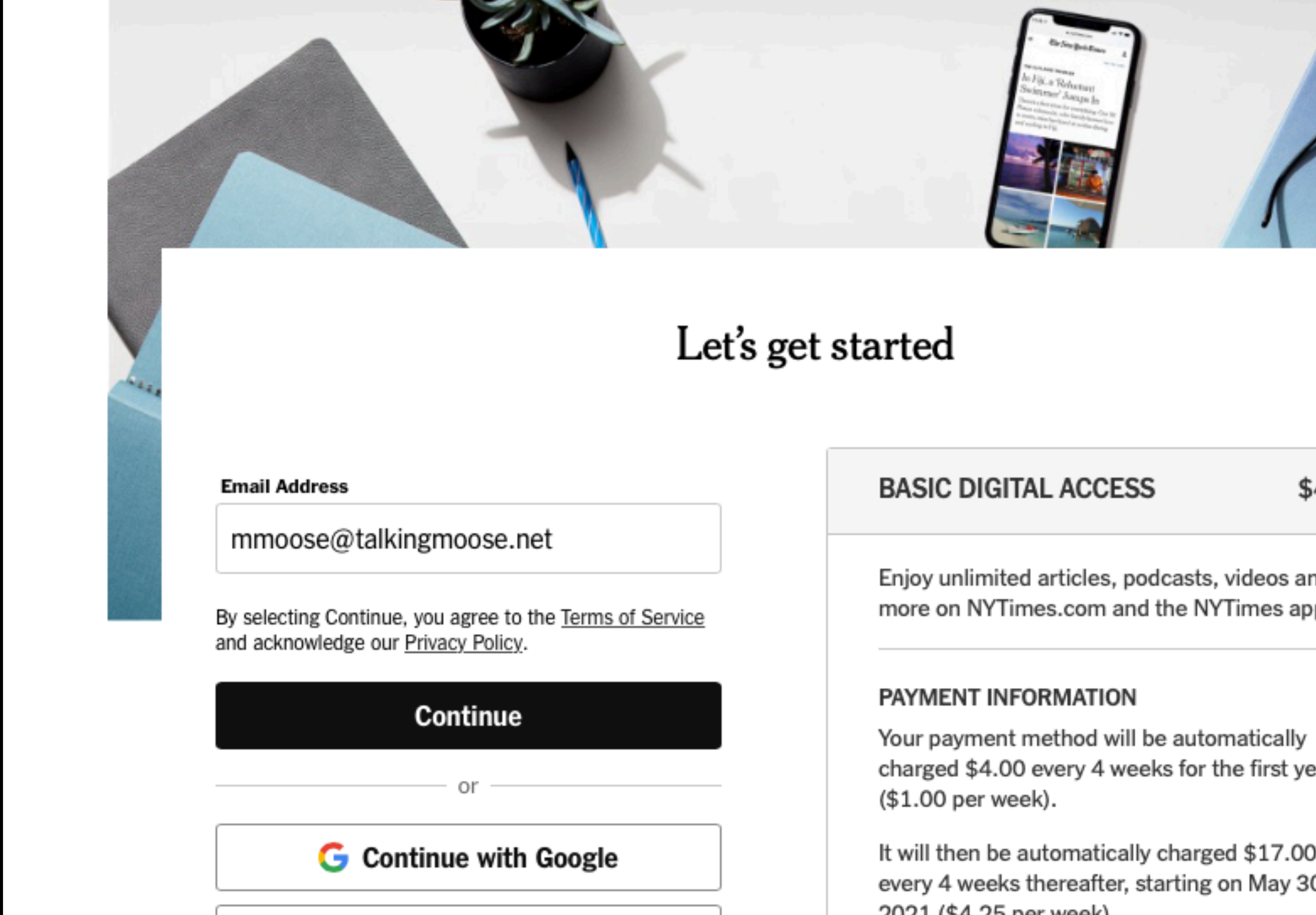
BASIC DIGITAL ACCESS $4

Enjoy unlimited articles, podcasts, videos an
more on NYTimes.com and the NYTimes ap

PAYMENT INFORMATION

Your payment method will be automatically
charged $4.00 every 4 weeks for the first ye
($1.00 per week).

It will then be automatically charged $17.00
every 4 weeks thereafter, starting on May 3
2021 ($4.25 per week).



Let's get started

Email Address

mmoose@talkingmoose.net

By selecting Continue, you agree to the Terms of Service and acknowledge our Privacy Policy.

BASIC DIGITAL ACCESS $

Enjoy unlimited articles, podcasts, videos an
more on NYTimes.com and the NYTimes ap

PAYMENT INFORMATION

Your payment method will be automatically
charged $4.00 every 4 weeks for the first ye
($1.00 per week).

It will then be automatically charged $17.00
every 4 weeks thereafter, starting on May 3
2021 ($4.25 per week).

mmoose@talkingmoose.net

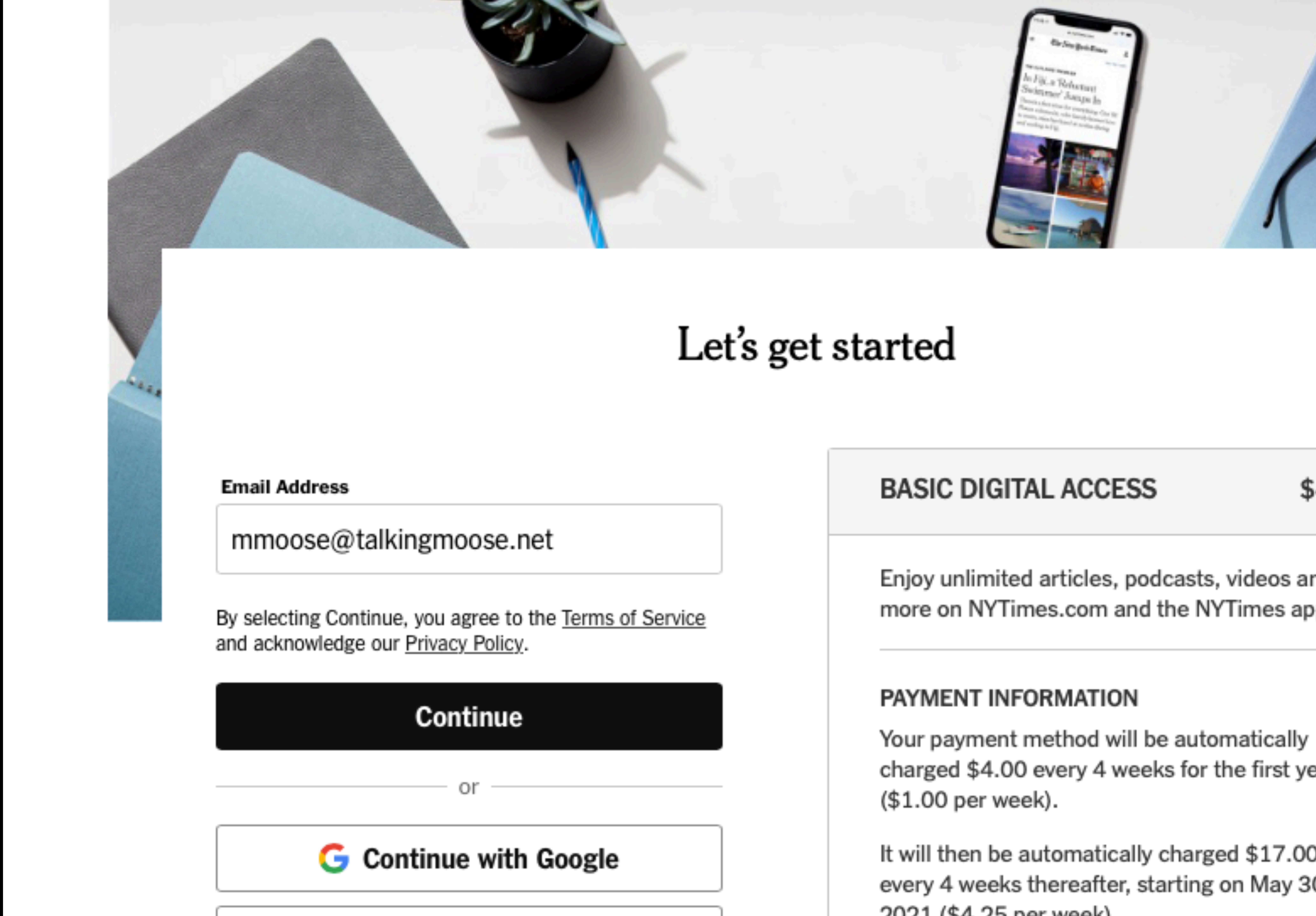
$$\underbrace{\text{mmoose}}_{\text{limit 64}} \quad @ \quad \underbrace{\text{talkingmoose.net}}_{\text{limit 255}}$$

56^{64}	37^{255}
7.656560673695E+111	infinity or wtf

$7.656560673695E+111^{\text{infinity or wtf}}$

GAME
OVER
PLAY AGAIN?

Save the internet. Use regex.



Let's get started

Email Address

mmoose@talkingmoose.net

By selecting Continue, you agree to the Terms of Service and acknowledge our Privacy Policy.

BASIC DIGITAL ACCESS $

Enjoy unlimited articles, podcasts, videos ar
more on NYTimes.com and the NYTimes ap

PAYMENT INFORMATION

Your payment method will be automatically
charged $4.00 every 4 weeks for the first ye
($1.00 per week).

It will then be automatically charged $17.00
every 4 weeks thereafter, starting on May 30
2021 ($4.25 per week)

$$\underbrace{\text{mmoose}}_{\text{limit 64}} \quad @ \quad \underbrace{\text{talkingmoose.net}}_{\text{limit 255}}$$

An email address is:

"A string of up to 64 characters…

followed by an @-symbol…

followed by a string of up to 255 characters…

with one of those characters being a dot somewhere in the middle."

```
\w{1,64}@\w{1,252}\.\w{2,253}
```

(regex to match an email address)

```
\w{1,64}@\w{1,252}\.\w{2,253}
```

(regex to match an email address)

```
\w{1,64}@\w{1,252}\.\w{2,253}
```

(regex to match an email address)

`\w` = any "word" character (a-z, A-Z or 0-9)

```
\w{1,64}@\w{1,252}\.\w{2,253}
```

(regex to match an email address)

`\w` = any “word” character (a-z, A-Z or 0-9)

```
\w{1,64}@\w{1,252}\.\w{2,253}
```

(regex to match an email address)

`\w` = any "word" character (a-z, A-Z or 0-9)
`{ }` = "occurrence indicator" or "repetition operator" (repeat the preceding character)

```
\w{1,64}@\w{1,252}\.\w{2,253}
```

(regex to match an email address)

`\w` = any "word" character (a-z, A-Z or 0-9)
`{ }` = "occurrence indicator" or "repetition operator" (repeat the preceding character)
`1,64` = match at least once or up to 64 times

`\w{1,64}` = mmoose

`\w{1,64}` = martin

`\w{1,64}` ≠ martin.moose

`\w` = any “word” character (a-z, A-Z or 0-9)

`{ }` = “occurrence indicator” or "repetition operator” (repeat the preceding character)

`1,64` = match at least once or up to 64 times

```
\w{1,64}@\w{1,252}\.\w{2,253}
```

(regex to match an email address)

`\w` = any "word" character (a-z, A-Z or 0-9)
`{ }` = "occurrence indicator" or "repetition operator" (repeat the preceding character)
`1,64` = match at least once or up to 64 times

```
\w{1,64}@\w{1,252}\.\w{2,253}
```

(regex to match an email address)

`\w` = any "word" character (a-z, A-Z or 0-9)

`{ }` = "occurrence indicator" or "repetition operator" (repeat the preceding character)

`1,64` = match at least once or up to 64 times

`@` = @

escape

`\.` = .

```
/usr/bin/osascript "display dialog "Hello World!" buttons {"OK"}"
```

```
/usr/bin/osascript "display dialog \"Hello World!\" buttons {\"OK\"}"
```

```
\w{1,64}@\w{1,252}\.\w{2,253}
```

(regex to match an email address)

`\w` = any "word" character (a-z, A-Z or 0-9)

`{ }` = "occurrence indicator" or "repetition operator" (repeat the preceding character)

`1,64` = match at least once or up to 64 times

`@` = @

escape $\rightarrow$ `\.` = .

Agenda

What is regex?

Characters with special meanings

Character classes and grouping

Applications and command line tools that support regex

Examples from real world experiences

Regex resources

Agenda

What is regex?

Characters with special meanings

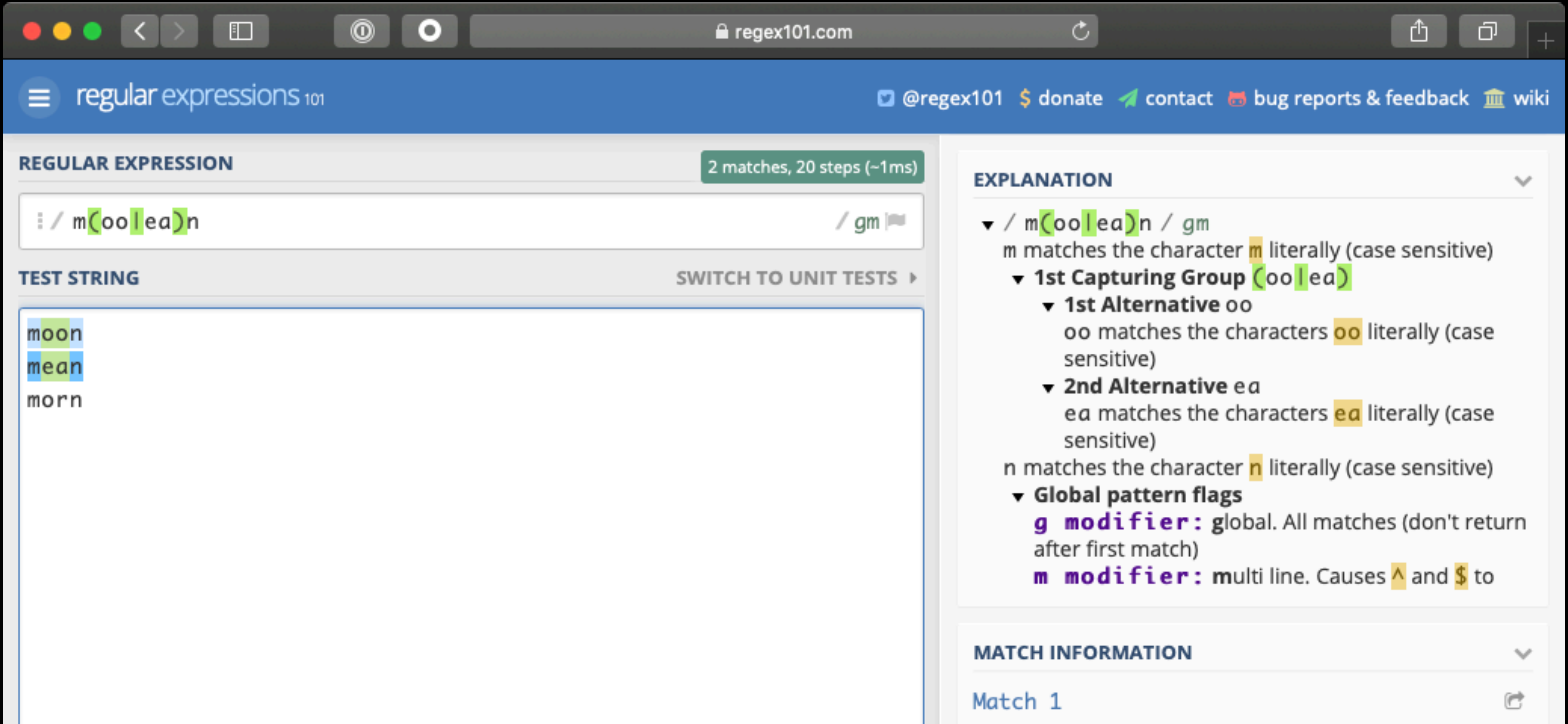
Character sets and grouping

Applications and command line tools that support regex

Examples from real world experiences

Regex resources

https://regex101.com



REGULAR EXPRESSION

2 matches, 20 steps (~1ms)

```
/ m(oo|ea)n / gm
```

TEST STRING

SWITCH TO UNIT TESTS ▸

```
moon
mean
morn
```

EXPLANATION

▾ / m(oo|ea)n / gm

m matches the character m literally (case sensitive)

▾ 1st Capturing Group (oo|ea)

▾ 1st Alternative oo

oo matches the characters oo literally (case sensitive)

▾ 2nd Alternative ea

ea matches the characters ea literally (case sensitive)

n matches the character n literally (case sensitive)

▾ Global pattern flags

g modifier: global. All matches (don't return after first match)

m modifier: multi line. Causes ^ and $ to

MATCH INFORMATION

Match 1

There is no shame in creating a regex cheatsheet.

There is no shame in creating a regex cheatsheet.

Letters and numbers match themselves

abc	=	abc
XYZ	=	XYZ
123	=	123
moon	=	moon
Moon	=	Moon
moon	≠	Moon
456	≠	564
abc	≠	ABC
Penn State	≠	PennState

A period
matches any character

.	=	a
.	=	A
.	=	1
.oon	=	Moon, moon, Loon, loon, toon
M..n	=	Moon, MOOn, Mean, M33n, M-sn
4..	=	456, 412, 4Ab
moon.	≠	moon123
Penn.State	≠	PennState
Penn.State	=	Penn State

a b c ...	=	*lowercase letters match themselves*
A B C ...	=	*UPPERcase letters match themselves*
1 2 3 ...	=	*numbers match themselves*

Square brackets indicate a choice of one character

[abc]	=	a, b or c
[XYZ]	=	X, Y or Z
[123]	=	1, 2 or 3

hat

[^aeiou]	=	not a, e, i, o nor u
[^RSTLNE]	=	not R, S, T, L, N nor E
[^024680]	=	not 0, 2, 4, 6 nor 8
[AaBbCc]	=	a, b, c, A, B or C
[Dog]	≠	Dog
D [Oo] [Gg]	=	Dog, DoG, DOG or DOg

a b c ...	=	*lower case letters match themselves*
A B C ...	=	*UPPER case letters match themselves*
1 2 3 ...	=	*numbers match themselves*
.	=	*any single character*
\\.	=	*period*

Square brackets support ranges of letters or numbers

[a-z]	=	any lower case letter a through z
[A-Z]	=	any upper case letter A through Z
[0-9]	=	any digit 0 through 9
[^a-e]	=	not a through e nor c
[^L-PX-Z]	=	not L through P, not X through Z
[1-489]	=	1 through 4, 8 or 9
[A-C1-3].	=	Az, B3, CQ, 1@, 24 or 3a
[a-Z]		invalid range
[a-9]		invalid range

a b c ...	=	*lower case letters match themselves*
A B C ...	=	*UPPER case letters match themselves*
1 2 3 ...	=	*numbers match themselves*
.	=	*any single character*
\\.	=	*period*
[a b c]	=	*match one of these characters*
[^ a b c]	=	*don't match any of these characters*

Repetitions and optional characters

`*`	=	repeat the preceding character 0 or more times
`q*`	=	q, qq, qqq, qqqq, etc., or no match
`+`	=	repeat the preceding character 1 or more times
`q+`	=	q, qq, qqq, qqqq, etc.
`{n}`	=	repeat the preceding character *n* times
`q{4}`	=	qqqq
`{m,n}`	=	repeat the preceding character *m* to *n* times
`q{2,4}`	=	qq, qqq or qqqq
`?`	=	the preceding character is optional
`q?`	=	q or no match

a b c ...	=	*lower case letters match themselves*
A B C ...	=	*UPPER case letters match themselves*
1 2 3 ...	=	*numbers match themselves*
.	=	*any single character*
\\.	=	*period*
[a b c]	=	*match one of these characters*
[^ a b c]	=	*don't match any of these characters*
[a - z]	=	*match any letter a through z*
[A - Z]	=	*match any letter A through Z*
[0 - 9]	=	*match any digit 0 through 9*

Repetitions and optional characters

5*-5*	=	555-5555, 5-5, -5, -
.*	=	The quick brown fox... or nothing
16\.1[7-9].*	=	16.17.1, 16.18, 16.19.543b3
5+-5+	=	555-5555, 5-5
0+7	=	07, 007, 0007
No{12}!	=	Noooooooooooo!
\d{3}-\d{4}	=	555-5555, 123-4567, 384-1717
<-{2,6}>	=	<-->, <--->, <---->, <-----> or <------>
colou?r	=	colour or color
alumini?um	=	aluminum

digit

a b c ...	=	*lower case letters match themselves*
A B C ...	=	*UPPER case letters match themselves*
1 2 3 ...	=	*numbers match themselves*
.	=	*any single character*
\ .	=	*period*
[a b c]	=	*match one of these characters*
[^ a b c]	=	*don't match any of these characters*
[a - z]	=	*match any letter a through z*
[A - Z]	=	*match any letter A through Z*
[0 - 9]	=	*match any digit 0 through 9*
*	=	*repeat the character 0 or more times*
+	=	*repeat the character 1 or more times*
{n}	=	*repeat the character n times*
{m,n}	=	*repeat the character m through n times*
?	=	*the character is optional*

Capture groups and the alternation operator

or

(abc)	=	abc
(IMG)?\d\.jpg	=	IMG2.jpg or 7.jpg
(ei){2}o	=	Old MacDonald had a farm...
(abc|123)	=	abc or 123
m(oo|ea)n	=	moon or mean
(1(0|1)){2}	=	1010 or 1011 or 1110 or 1111

a b c ...	=	*lower case letters match themselves*
A B C ...	=	*UPPER case letters match themselves*
1 2 3 ...	=	*numbers match themselves*
.	=	*any single character*
\\.	=	*period*
[a b c]	=	*match one of these characters*
[^ a b c]	=	*don't match any of these characters*
[a - z]	=	*match any letter a through z*
[A - Z]	=	*match any letter A through Z*
[0 - 9]	=	*match any digit 0 through 9*
*	=	*repeat the character 0 or more times*
+	=	*repeat the character 1 or more times*
{n}	=	*repeat the character n times*
{m,n}	=	*repeat the character m through n times*
?	=	*the character is optional*
\w	=	*match any letter or number*
\d	=	*match any digit*
\D	=	*match any non-digit character*

Building regexes

MacBookAir8,2	MacBookAir7,1	MacBookAir6,1
MacBookAir8,1	MacBookAir6,2	MacBookAir5,2
MacBookAir7,2	MacBookAir6,1	MacBookAir5,1
MacBookAir7,2	MacBookAir6,2	

```
MacBookAir[5-8],[12]
MacBookAir[5-8],(1|2)
```

a b c ...	=	*lower case letters match themselves*
A B C ...	=	*UPPER case letters match themselves*
1 2 3 ...	=	*numbers match themselves*
.	=	*any single character*
\\.	=	*period*
[a b c]	=	*match one of these characters*
[^ a b c]	=	*don't match any of these characters*
[a - z]	=	*match any letter a through z*
[A - Z]	=	*match any letter A through Z*
[0 - 9]	=	*match any digit 0 through 9*
*	=	*repeat the character 0 or more times*
+	=	*repeat the character 1 or more times*
{*n*}	=	*repeat the character n times*
{*m*,*n*}	=	*repeat the character m through n times*
?	=	*the character is optional*
\w	=	*match any letter or number*
\d	=	*match any digit*
\D	=	*match any non-digit character*
(abc)	=	*match the string in parentheses*
(a | b | c)	=	*or*

Building regexes

10.11.0.99	10.11.0.25	10.20.0.5
10.11.0.100	10.11.0.50	10.20.0.40
10.11.0.132	10.11.0.150	10.20.0.122
10.11.0.200	10.11.0.200	10.20.0.179

👍 `10\.(11|20)\.0\.\d{1,3}`

000 - 999

a b c ...	=	*lower case letters match themselves*
A B C ...	=	*UPPER case letters match themselves*
1 2 3 ...	=	*numbers match themselves*
.	=	*any single character*
\\.	=	*period*
[a b c]	=	*match one of these characters*
[^ a b c]	=	*don't match any of these characters*
[a - z]	=	*match any letter a through z*
[A - Z]	=	*match any letter A through Z*
[0 - 9]	=	*match any digit 0 through 9*
*	=	*repeat the character 0 or more times*
+	=	*repeat the character 1 or more times*
{n}	=	*repeat the character n times*
{m,n}	=	*repeat the character m through n times*
?	=	*the character is optional*
\w	=	*match any letter or number*
\d	=	*match any digit*
\D	=	*match any non-digit character*
(abc)	=	*match the string in parentheses*
(a | b | c)	=	*or*

Building regexes

16.17	16.20.1	16.23.1
16.18	16.22	16.24
16.19	16.22.1	16.24.1
16.20	16.23	*to 16.52*

```
16\.(1[7-9]|[2-4][0-9]|5[0-2]).*
```

17-19 20-49 50-52

a b c ...	=	*lower case letters match themselves*
A B C ...	=	*UPPER case letters match themselves*
1 2 3 ...	=	*numbers match themselves*
.	=	*any single character*
\\.	=	*period*
[a b c]	=	*match one of these characters*
[^ a b c]	=	*don't match any of these characters*
[a - z]	=	*match any letter a through z*
[A - Z]	=	*match any letter A through Z*
[0 - 9]	=	*match any digit 0 through 9*
*	=	*repeat the character 0 or more times*
+	=	*repeat the character 1 or more times*
{*n*}	=	*repeat the character n times*
{*m*,*n*}	=	*repeat the character m through n times*
?	=	*the character is optional*
\w	=	*match any letter or number*
\d	=	*match any digit*
\D	=	*match any non-digit character*
(abc)	=	*match the string in parentheses*
(a | b | c)	=	*or*

Agenda

What is regex?

Characters with special meanings

Character sets and grouping

Applications and command line tools that support regex

Examples from real world experiences

Regex resources

Agenda

What is regex?

Characters with special meanings

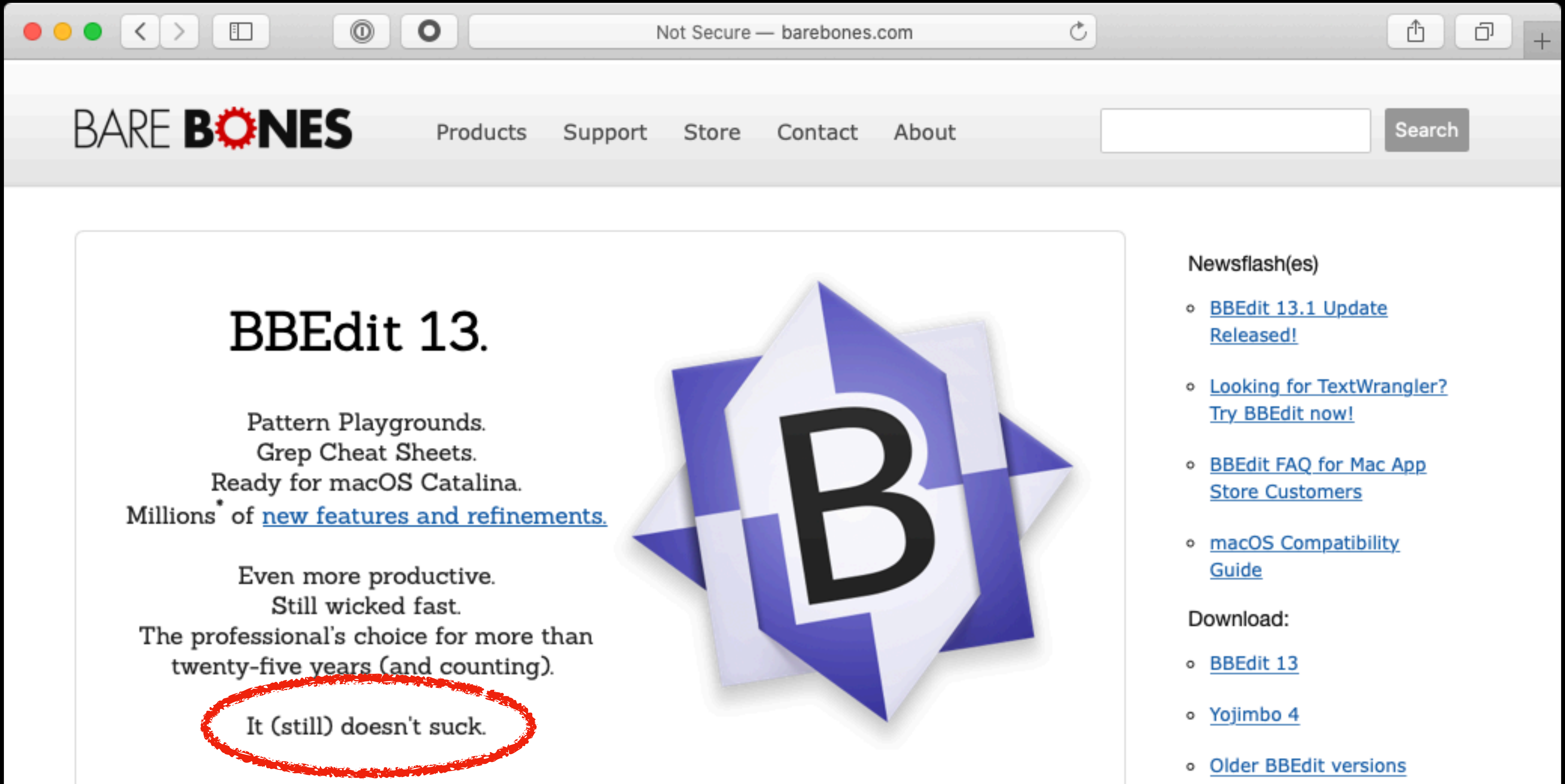
Character sets and grouping

Applications and command line tools that support regex

Examples from real world experiences

Regex resources

BBEdit from https://barebones.com



BBEdit 13.

Pattern Playgrounds.
Grep Cheat Sheets.
Ready for macOS Catalina.
Millions* of new features and refinements.

Even more productive.
Still wicked fast.
The professional's choice for more than twenty-five years (and counting).

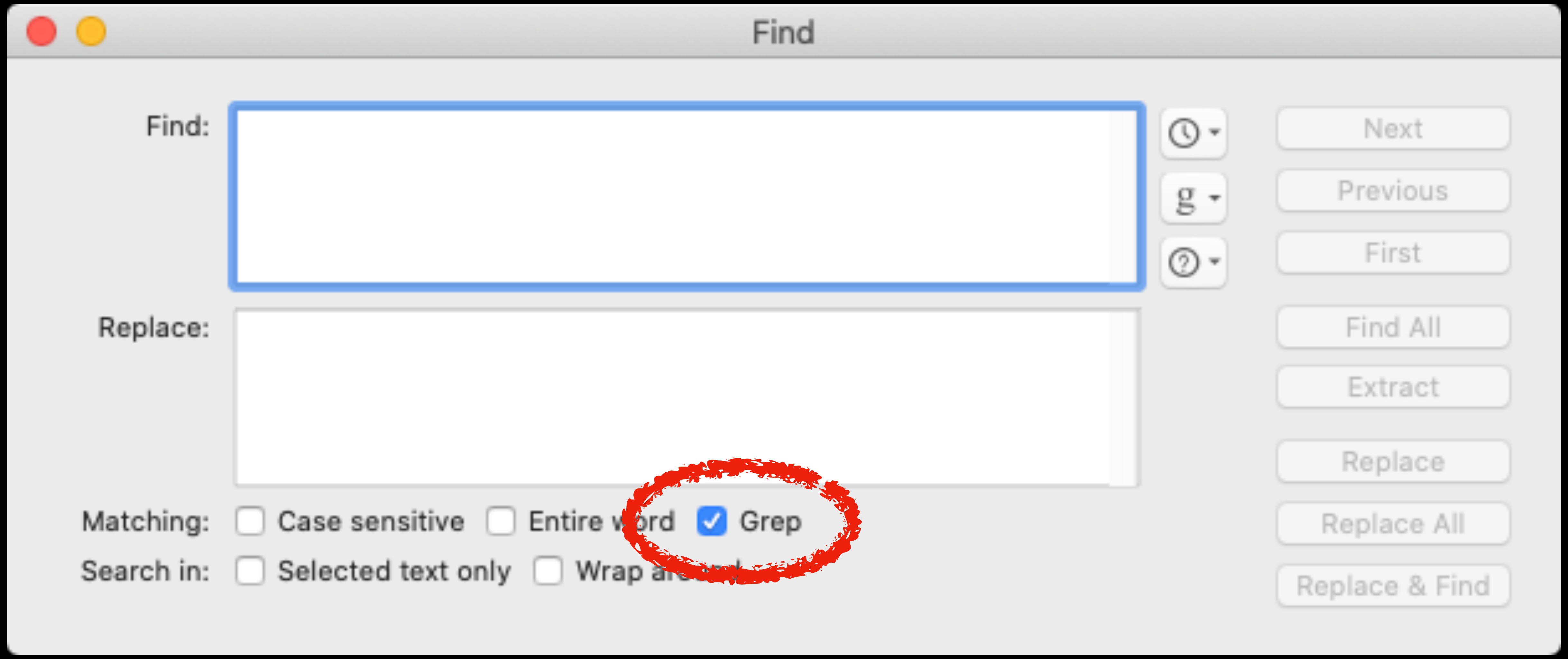
It (still) doesn't suck.

Newsflash(es)

- BBEdit 13.1 Update Released!
- Looking for TextWrangler? Try BBEdit now!
- BBEdit FAQ for Mac App Store Customers
- macOS Compatibility Guide

Download:

- BBEdit 13
- Yojimbo 4
- Older BBEdit versions



Matching: [ ] Case sensitive [ ] Entire word [✓ Grep]

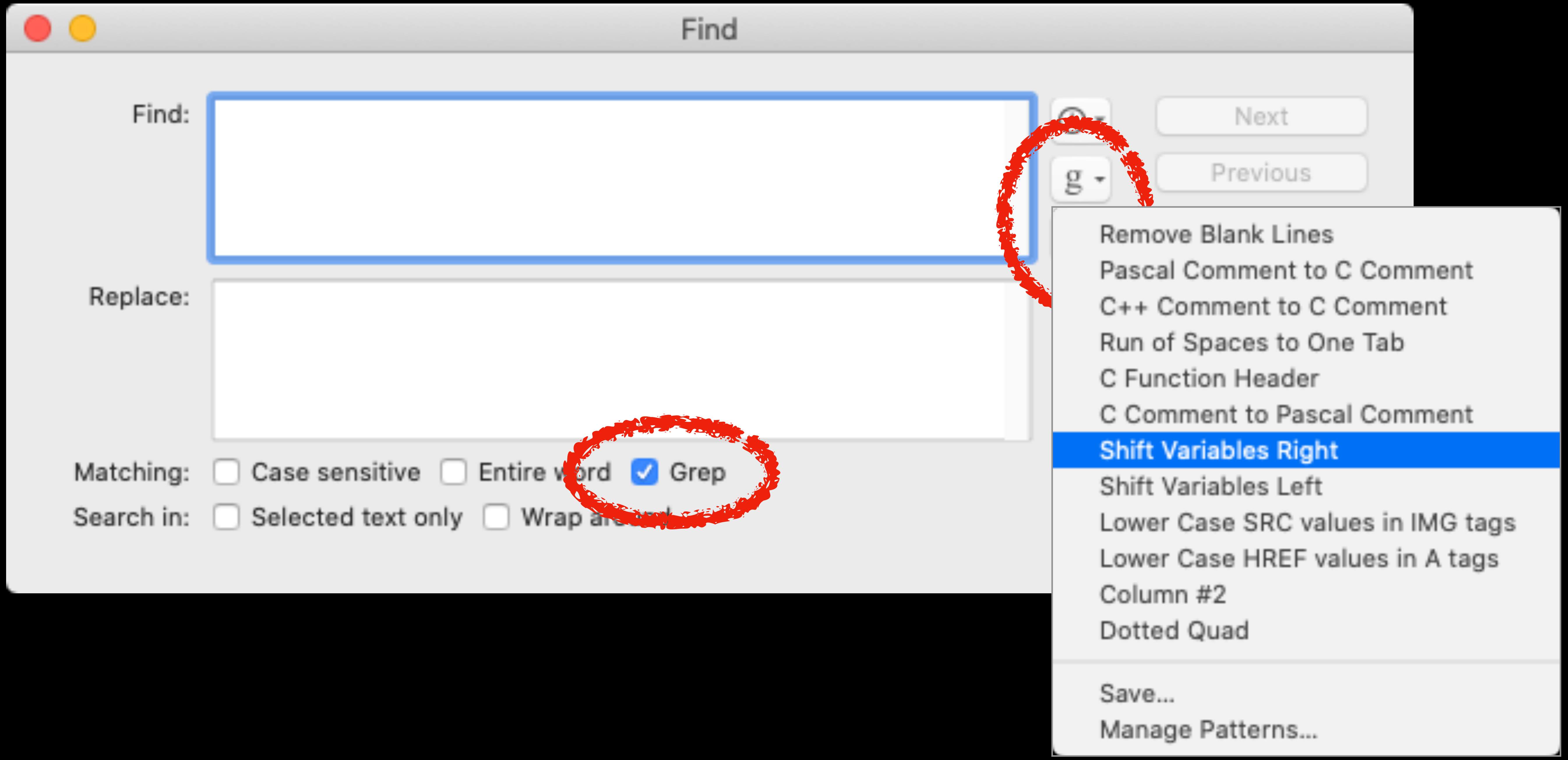
Search in: [ ] Selected text only [ ] Wrap around

useless use of "cat"

```
cat file.txt | grep dog
```

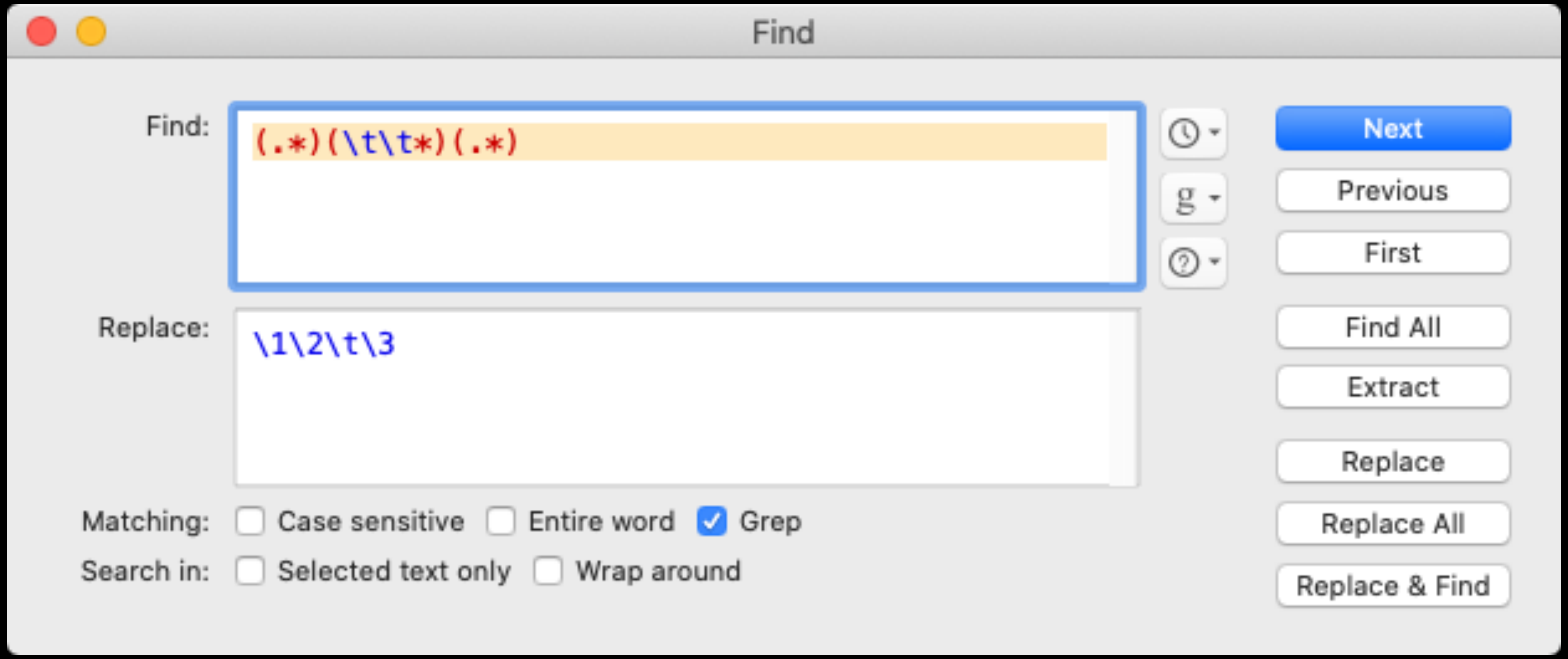
grep

(Global Regular Expression Print)



Matching: [ ] Case sensitive [ ] Entire word [✓ Grep]

Search in: [ ] Selected text only [ ] Wrap around



Find:

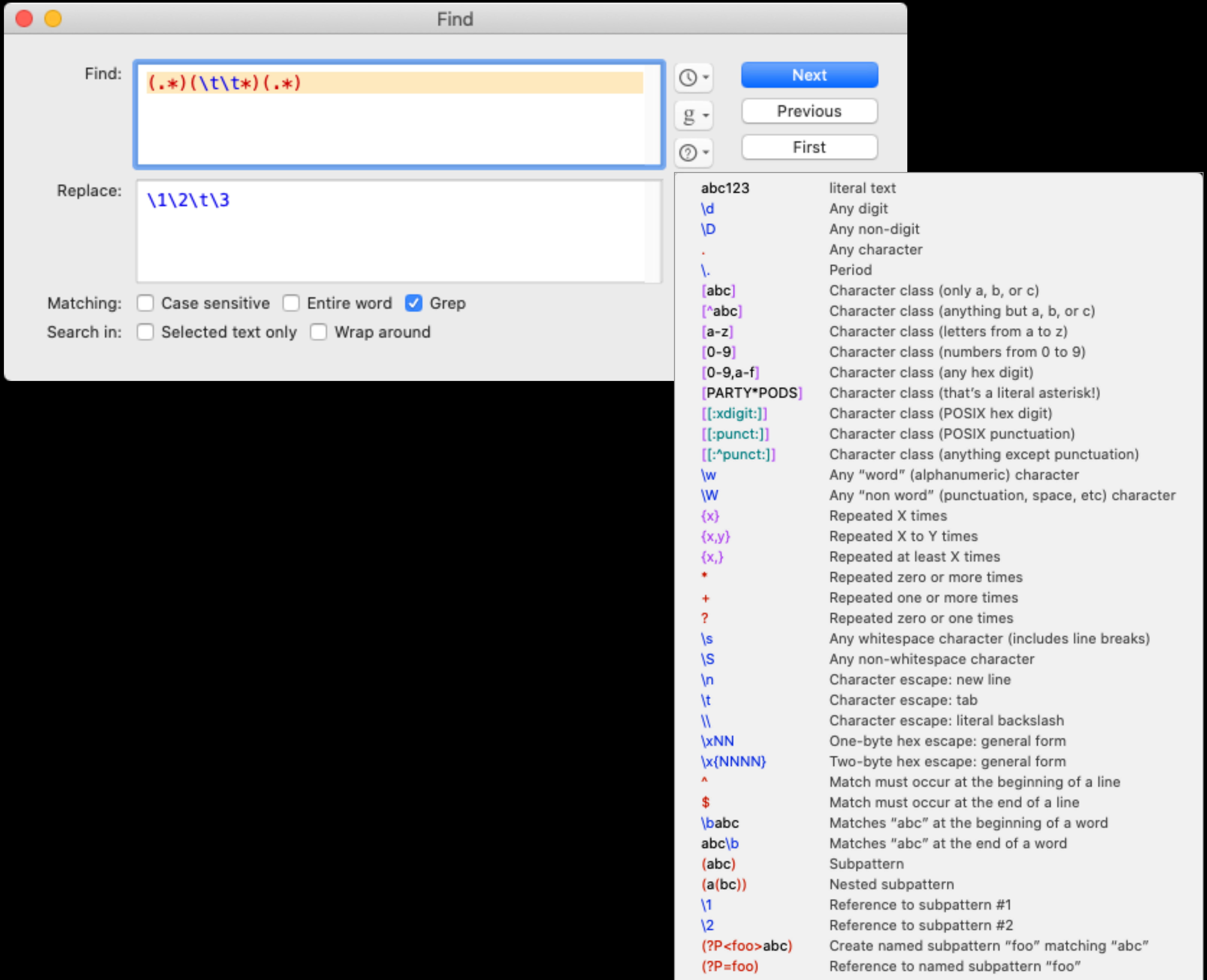
```
(.*)(\t\t*)(.*)
```

Replace:

```
\1\2\t\3
```

Matching: [ ] Case sensitive [ ] Entire word [✓ Grep]

Search in: [ ] Selected text only [ ] Wrap around



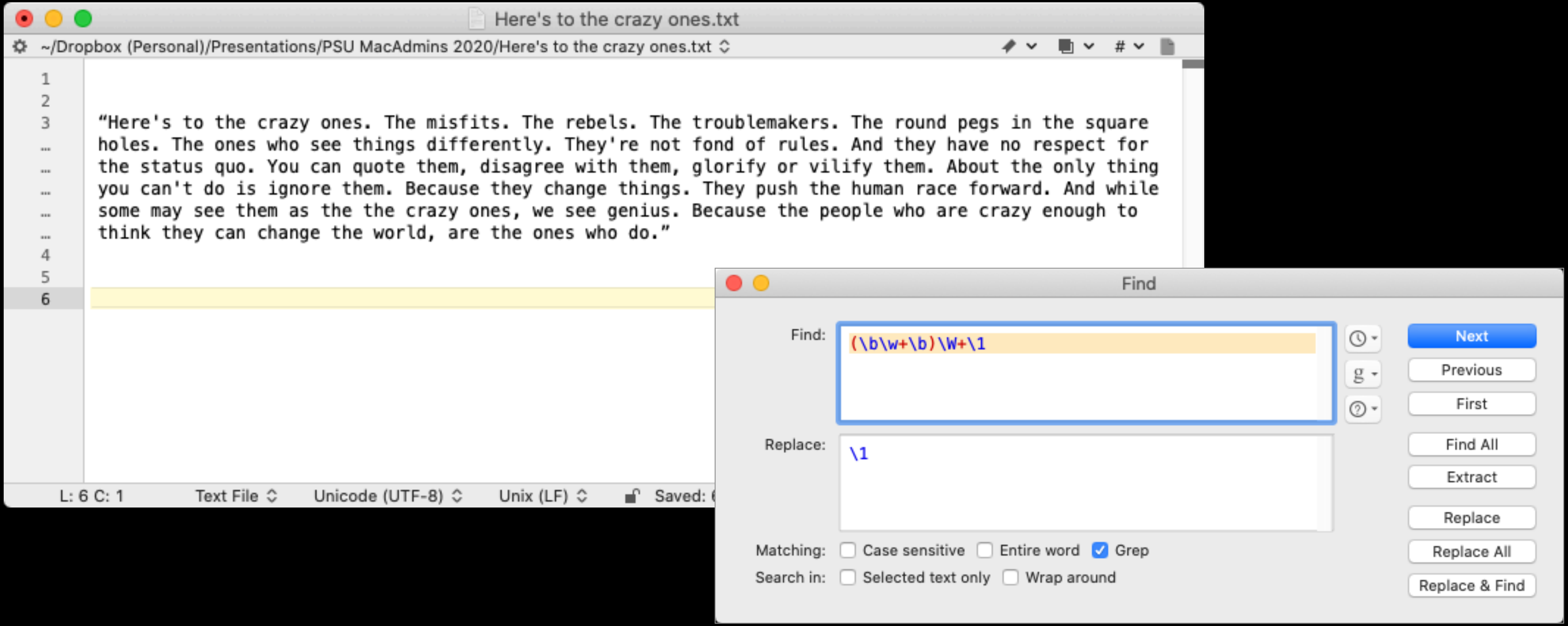
Find: `(.*)(\t\t*)(.*)`

Replace: `\1\2\t\3`

Matching: [ ] Case sensitive [ ] Entire word [✓ Grep]

Search in: [ ] Selected text only [ ] Wrap around

abc123	literal text
\d	Any digit
\D	Any non-digit
.	Any character
\\.	Period
[abc]	Character class (only a, b, or c)
[^abc]	Character class (anything but a, b, or c)
[a-z]	Character class (letters from a to z)
[0-9]	Character class (numbers from 0 to 9)
[0-9,a-f]	Character class (any hex digit)
[PARTY*PODS]	Character class (that's a literal asterisk!)
[[:xdigit:]]	Character class (POSIX hex digit)
[[:punct:]]	Character class (POSIX punctuation)
[[:^punct:]]	Character class (anything except punctuation)
\w	Any "word" (alphanumeric) character
\W	Any "non word" (punctuation, space, etc) character
{x}	Repeated X times
{x,y}	Repeated X to Y times
{x,}	Repeated at least X times
*	Repeated zero or more times
+	Repeated one or more times
?	Repeated zero or one times
\s	Any whitespace character (includes line breaks)
\S	Any non-whitespace character
\n	Character escape: new line
\t	Character escape: tab
\\\\	Character escape: literal backslash
\xNN	One-byte hex escape: general form
\x{NNNN}	Two-byte hex escape: general form
^	Match must occur at the beginning of a line
$	Match must occur at the end of a line
\babc	Matches "abc" at the beginning of a word
abc\b	Matches "abc" at the end of a word
(abc)	Subpattern
(a(bc))	Nested subpattern
\1	Reference to subpattern #1
\2	Reference to subpattern #2
(?P<foo>abc)	Create named subpattern "foo" matching "abc"
(?P=foo)	Reference to named subpattern "foo"


Here's to the crazy ones.txt
~/Dropbox (Personal)/Presentations/PSU MacAdmins 2020/Here's to the crazy ones.txt

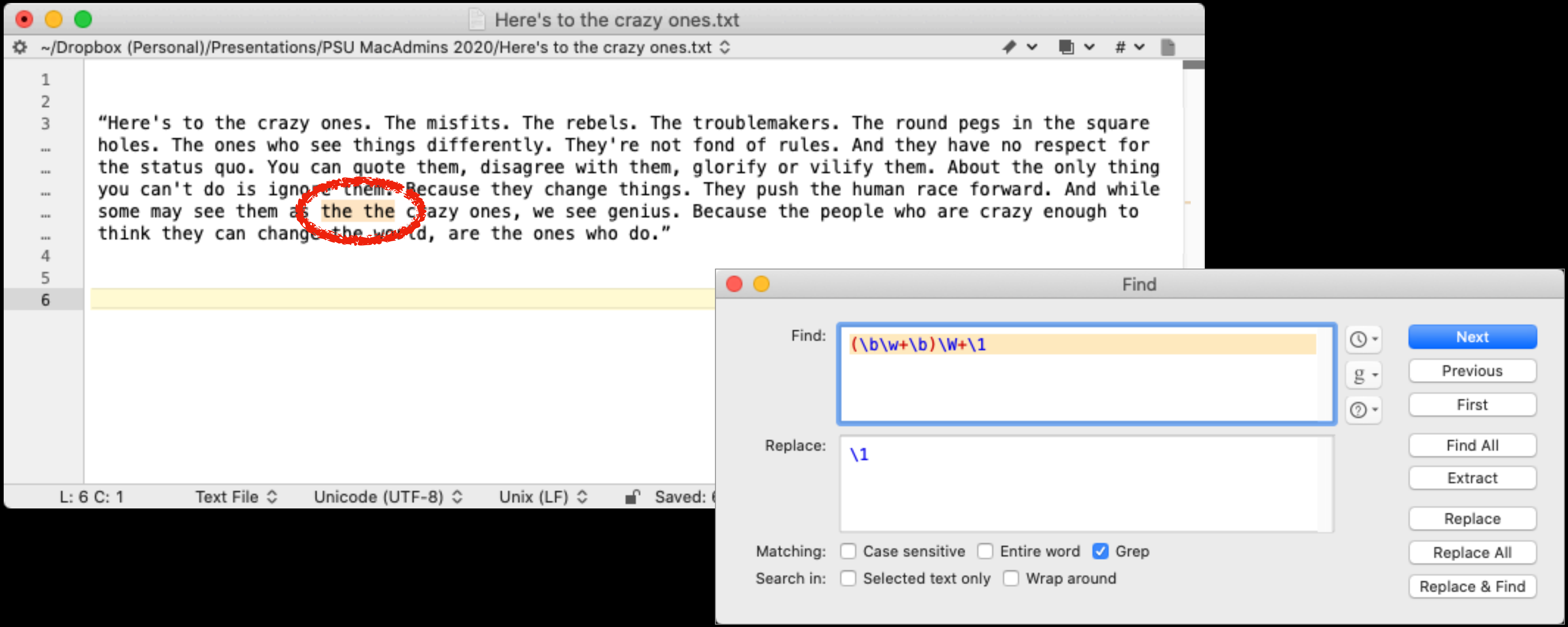
```
“Here's to the crazy ones. The misfits. The rebels. The troublemakers. The round pegs in the square
holes. The ones who see things differently. They're not fond of rules. And they have no respect for
the status quo. You can quote them, disagree with them, glorify or vilify them. About the only thing
you can't do is ignore them. Because they change things. They push the human race forward. And while
some may see them as the the crazy ones, we see genius. Because the people who are crazy enough to
think they can change the world, are the ones who do.”
```

Find: `(\b\w+\b)\W+\1`

Replace: `\1`

Matching: [ ] Case sensitive [ ] Entire word [✓ Grep]

Search in: [ ] Selected text only [ ] Wrap around


Here's to the crazy ones.txt
~/Dropbox (Personal)/Presentations/PSU MacAdmins 2020/Here's to the crazy ones.txt

“Here's to the crazy ones. The misfits. The rebels. The troublemakers. The round pegs in the square holes. The ones who see things differently. They're not fond of rules. And they have no respect for the status quo. You can quote them, disagree with them, glorify or vilify them. About the only thing you can't do is ignore them. Because they change things. They push the human race forward. And while some may see them as the the crazy ones, we see genius. Because the people who are crazy enough to think they can change the world, are the ones who do.”

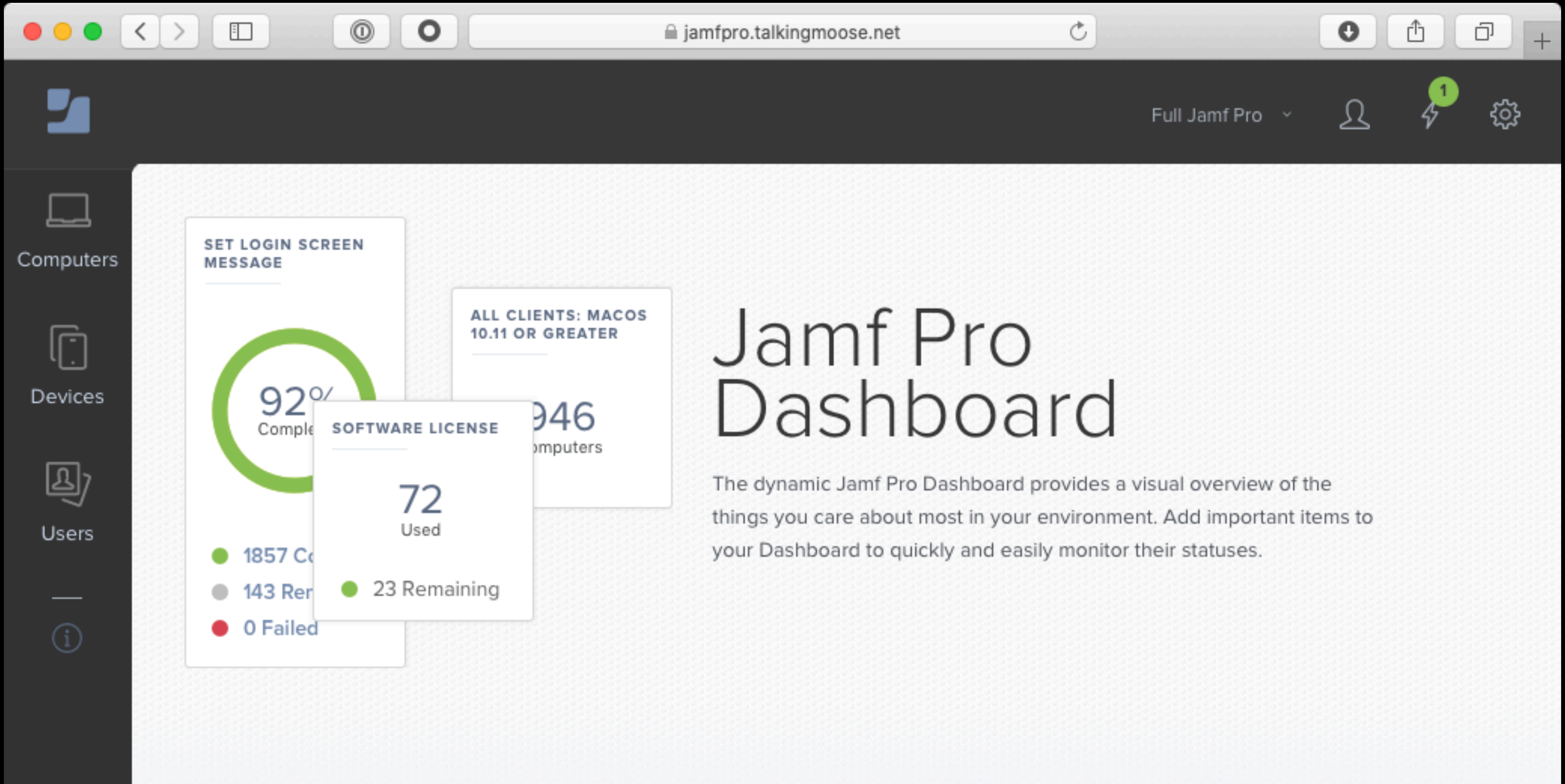
Find: `(\b\w+\b)\W+\1`

Replace: `\1`

Matching: [ ] Case sensitive [ ] Entire word [✓ Grep]

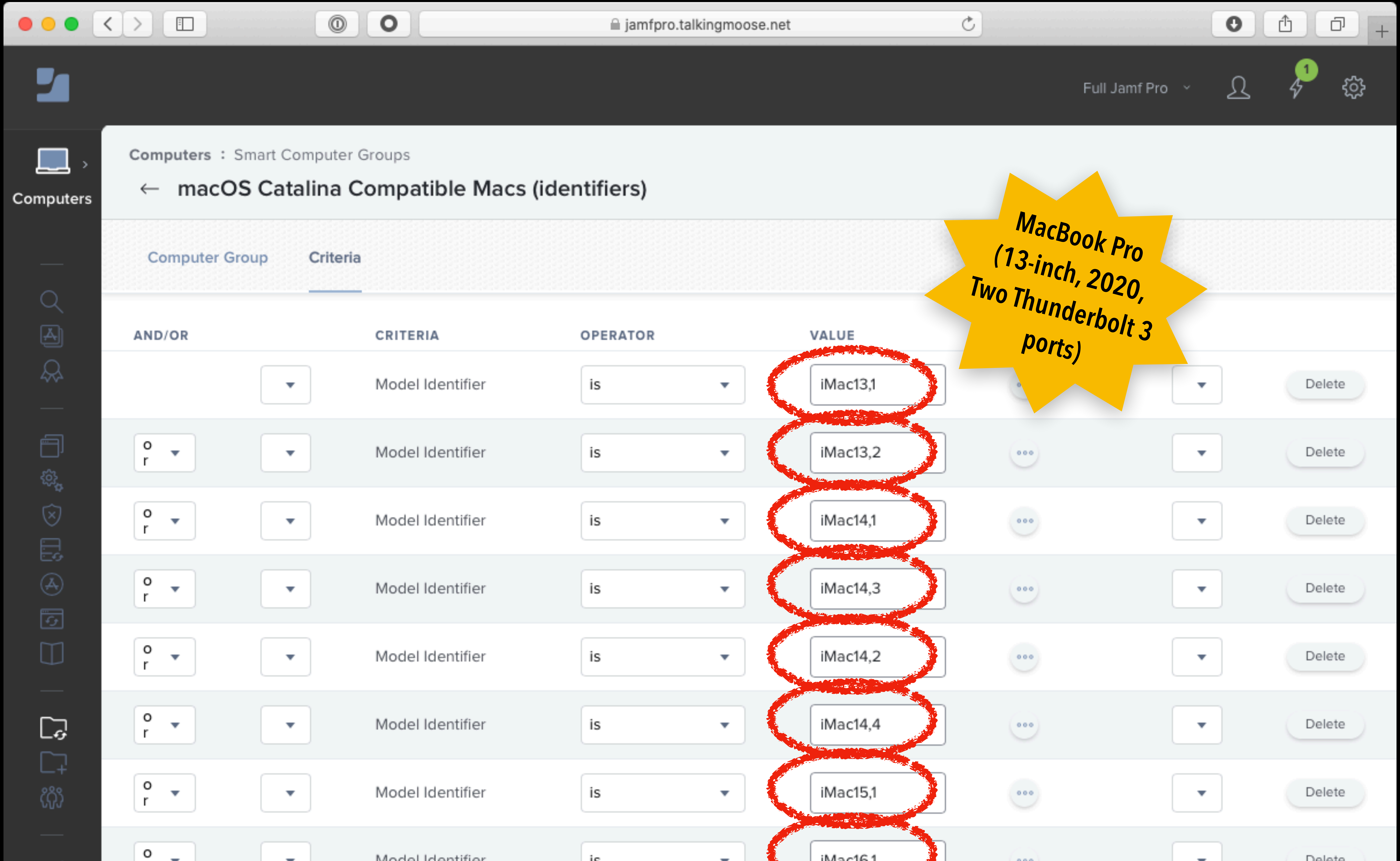
Search in: [ ] Selected text only [ ] Wrap around

Jamf Pro from https://www.jamf.com



Jamf Pro Dashboard

The dynamic Jamf Pro Dashboard provides a visual overview of the things you care about most in your environment. Add important items to your Dashboard to quickly and easily monitor their statuses.



jamfpro.talkingmoose.net

Computers : Smart Computer Groups

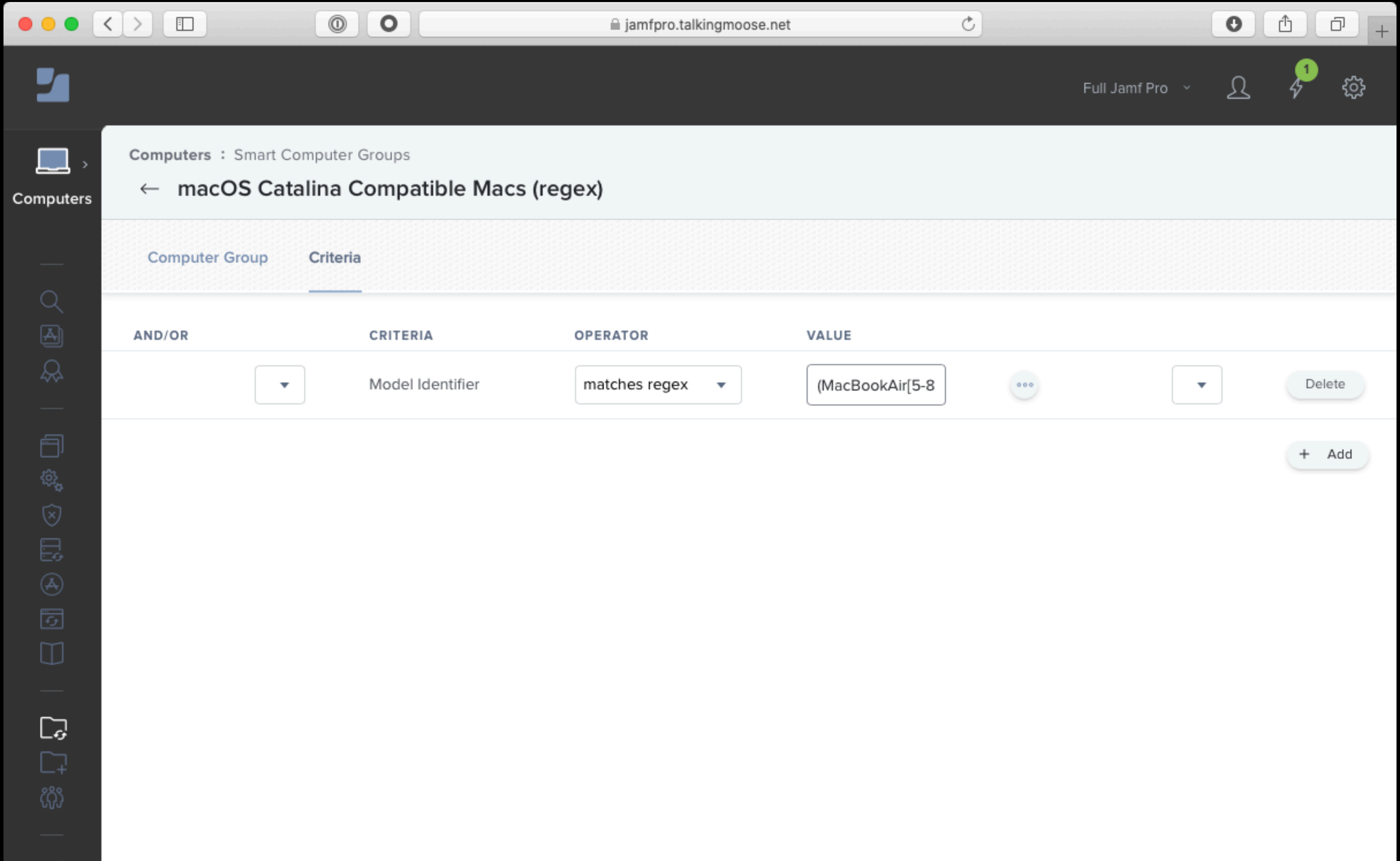
macOS Catalina Compatible Macs (identifiers)

AND/OR	CRITERIA	OPERATOR	VALUE
	Model Identifier	is	iMac13,1
or	Model Identifier	is	iMac13,2
or	Model Identifier	is	iMac14,1
or	Model Identifier	is	iMac14,3
or	Model Identifier	is	iMac14,2
or	Model Identifier	is	iMac14,4
or	Model Identifier	is	iMac15,1
o	Model Identifier	is	iMac16,1

```
(MacBookAir[5-9]|MacBookPro(9|1[0-6])|MacPro[6-7]|iMac(Pro)?1[3-9]?|MacBook(10|9|8)|Macmini[6-8]),\d
```

62 Model Identifiers

iMac13,1	iMac19,1	MacBookAir7,2	MacBookPro11,3	MacBookPro15,4
iMac13,2	iMac19,2	MacBookAir7,2	MacBookPro11,4	MacBookPro16,1
iMac14,1	iMacPro1,1	MacBookAir8,1	MacBookPro11,5	MacBookPro16,2
iMac14,2	MacBook8,1	MacBookAir8,2	MacBookPro12,1	MacBookPro16,3
iMac14,3	MacBook9,1	MacBookAir9,1	MacBookPro13,1	Macmini6,1
iMac14,4	MacBook10,1	MacBookPro9,1	MacBookPro13,2	Macmini6,2
iMac15,1	MacBookAir5,1	MacBookPro9,2	MacBookPro13,3	Macmini7,1
iMac16,1	MacBookAir5,2	MacBookPro10,1	MacBookPro14,1	Macmini8,1
iMac16,2	MacBookAir6,1	MacBookPro10,1	MacBookPro14,2	MacPro6,1
iMac17,1	MacBookAir6,1	MacBookPro10,2	MacBookPro14,3	MacPro7,1
iMac18,1	MacBookAir6,2	MacBookPro11,1	MacBookPro15,1	
iMac18,2	MacBookAir6,2	MacBookPro11,1	MacBookPro15,2	
iMac18,3	MacBookAir7,1	MacBookPro11,2	MacBookPro15,3	


jamfpro.talkingmoose.net

Full Jamf Pro

Computers

Computers : Smart Computer Groups

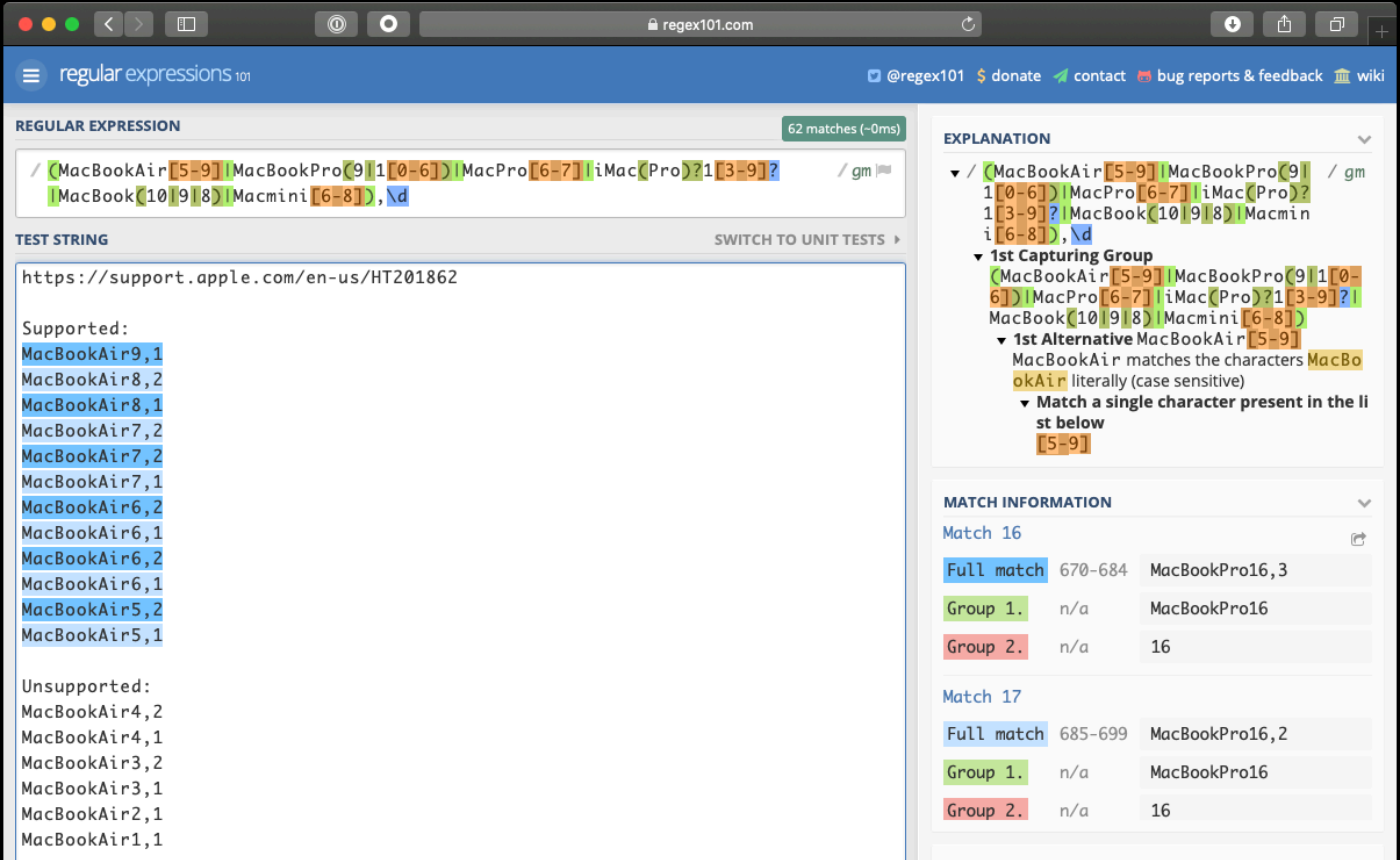
macOS Catalina Compatible Macs (regex)

Computer Group | Criteria

AND/OR	CRITERIA	OPERATOR	VALUE
	Model Identifier	matches regex	(MacBookAir[5-8

Delete

+ Add


regex101.com

regular expressions 101

@regex101 $ donate contact bug reports & feedback wiki

REGULAR EXPRESSION

62 matches (~0ms)

```
/ (MacBookAir[5-9]|MacBookPro(9|1[0-6])|MacPro[6-7]|iMac(Pro)?1[3-9]?
|MacBook(10|9|8)|Macmini[6-8]),\d                                      / gm
```

TEST STRING

SWITCH TO UNIT TESTS ›

```
https://support.apple.com/en-us/HT201862

Supported:
MacBookAir9,1
MacBookAir8,2
MacBookAir8,1
MacBookAir7,2
MacBookAir7,2
MacBookAir7,1
MacBookAir6,2
MacBookAir6,1
MacBookAir6,2
MacBookAir6,1
MacBookAir5,2
MacBookAir5,1

Unsupported:
MacBookAir4,2
MacBookAir4,1
MacBookAir3,2
MacBookAir3,1
MacBookAir2,1
MacBookAir1,1
```

EXPLANATION

```
▾ / (MacBookAir[5-9]|MacBookPro(9|1[0-6])|MacPro[6-7]|iMac(Pro)?1[3-9]?|MacBook(10|9|8)|Macmini[6-8]),\d / gm
  ▾ 1st Capturing Group (MacBookAir[5-9]|MacBookPro(9|1[0-6])|MacPro[6-7]|iMac(Pro)?1[3-9]?|MacBook(10|9|8)|Macmini[6-8])
    ▾ 1st Alternative MacBookAir[5-9]
      MacBookAir matches the characters MacBookAir literally (case sensitive)
      ▾ Match a single character present in the list below
        [5-9]
```

MATCH INFORMATION

Match 16

Full match	670-684	MacBookPro16,3
Group 1.	n/a	MacBookPro16
Group 2.	n/a	16

Match 17

Full match	685-699	MacBookPro16,2
Group 1.	n/a	MacBookPro16
Group 2.	n/a	16

grep

```
grep "Pro" model-identifiers.txt
```

```
iMac13,1      iMac19,1         MacBookAir7,2      MacBookPro11,3    MacBookPro15,4
iMac13,2      iMac19,2         MacBookAir7,2      MacBookPro11,4    MacBookPro16,1
iMac14,1      iMacPro1,1       MacBookAir8,1      MacBookPro11,5    MacBookPro16,2
iMac14,2      MacBook8,1       MacBookAir8,2      MacBookPro12,1    MacBookPro16,3
iMac14,3      MacBook9,1       MacBookAir9,1      MacBookPro13,1    Macmini6,1
iMac14,4      MacBook10,1      MacBookPro9,1      MacBookPro13,2    Macmini6,2
iMac15,1      MacBookAir5,1    MacBookPro9,2      MacBookPro13,3    Macmini7,1
iMac16,1      MacBookAir5,2    MacBookPro10,1     MacBookPro14,1    Macmini8,1
iMac16,2      MacBookAir6,1    MacBookPro10,1     MacBookPro14,2    MacPro6,1
iMac17,1      MacBookAir6,1    MacBookPro10,2     MacBookPro14,3    MacPro7,1
iMac18,1      MacBookAir6,2    MacBookPro11,1     MacBookPro15,1
iMac18,2      MacBookAir6,2    MacBookPro11,1     MacBookPro15,2
iMac18,3      MacBookAir7,1    MacBookPro11,2     MacBookPro15,3
```

grep

```
grep "Pro" model-identifiers.txt
```

iMac13,1	iMac19,1	MacBookAir7,2	**MacBookPro11,3**	**MacBookPro15,4**
iMac13,2	iMac19,2	MacBookAir7,2	**MacBookPro11,4**	**MacBookPro16,1**
iMac14,1	**iMacPro1,1**	MacBookAir8,1	**MacBookPro11,5**	**MacBookPro16,2**
iMac14,2	MacBook8,1	MacBookAir8,2	**MacBookPro12,1**	**MacBookPro16,3**
iMac14,3	MacBook9,1	MacBookAir9,1	**MacBookPro13,1**	Macmini6,1
iMac14,4	MacBook10,1	**MacBookPro9,1**	**MacBookPro13,2**	Macmini6,2
iMac15,1	MacBookAir5,1	**MacBookPro9,2**	**MacBookPro13,3**	Macmini7,1
iMac16,1	MacBookAir5,2	**MacBookPro10,1**	**MacBookPro14,1**	Macmini8,1
iMac16,2	MacBookAir6,1	**MacBookPro10,1**	**MacBookPro14,2**	**MacPro6,1**
iMac17,1	MacBookAir6,1	**MacBookPro10,2**	**MacBookPro14,3**	**MacPro7,1**
iMac18,1	MacBookAir6,2	**MacBookPro11,1**	**MacBookPro15,1**	
iMac18,2	MacBookAir6,2	**MacBookPro11,1**	**MacBookPro15,2**	
iMac18,3	MacBookAir7,1	**MacBookPro11,2**	**MacBookPro15,3**	

grep

```
grep "Pro" model-identifiers.txt
```

```
iMacPro1,1          MacBookPro13,1      MacPro6,1
MacBookPro9,1       MacBookPro13,2      MacPro7,1
MacBookPro9,2       MacBookPro13,3
MacBookPro10,1      MacBookPro14,1
MacBookPro10,1      MacBookPro14,2
MacBookPro10,2      MacBookPro14,3
MacBookPro11,1      MacBookPro15,1
MacBookPro11,1      MacBookPro15,2
MacBookPro11,2      MacBookPro15,3
MacBookPro11,3      MacBookPro15,4
MacBookPro11,4      MacBookPro16,1
MacBookPro11,5      MacBookPro16,2
MacBookPro12,1      MacBookPro16,3
```

grep

```
grep "Pro1[2-6]," model-identifiers.txt
```

```
iMacPro1,1          MacBookPro13,1      MacPro6,1
MacBookPro9,1       MacBookPro13,2      MacPro7,1
MacBookPro9,2       MacBookPro13,3
MacBookPro10,1      MacBookPro14,1
MacBookPro10,1      MacBookPro14,2
MacBookPro10,2      MacBookPro14,3
MacBookPro11,1      MacBookPro15,1
MacBookPro11,1      MacBookPro15,2
MacBookPro11,2      MacBookPro15,3
MacBookPro11,3      MacBookPro15,4
MacBookPro11,4      MacBookPro16,1
MacBookPro11,5      MacBookPro16,2
MacBookPro12,1      MacBookPro16,3
```

grep

```
grep "Pro1[2-6]," model-identifiers.txt
```

```
MacBookPro12,1          MacBookPro16,3
MacBookPro13,1
MacBookPro13,2
MacBookPro13,3
MacBookPro14,1
MacBookPro14,2
MacBookPro14,3
MacBookPro15,1
MacBookPro15,2
MacBookPro15,3
MacBookPro15,4
MacBookPro16,1
MacBookPro16,2
```

grep

```
grep "Pro(1[2-6]|\d)," model-identifiers.txt
```

grep

```
grep -E
grep --extended-regexp
```

(+, ?, |, (,))

grep -E

```
grep "Pro(1[2-6]|\d)," model-identifiers.txt
```

grep -E

```
grep -E "Pro(1[2-6]|\d)," model-identifiers.txt
```

```
iMacPro1,1            MacBookPro16,2
MacBookPro12,1        MacBookPro16,3
MacBookPro13,1        MacPro6,1
MacBookPro13,2        MacPro7,1
MacBookPro13,3
MacBookPro14,1
MacBookPro14,2
MacBookPro14,3
MacBookPro15,1
MacBookPro15,2
MacBookPro15,3
MacBookPro15,4
MacBookPro16,1
```

grep -E

```
grep -E  =  egrep
```

awk

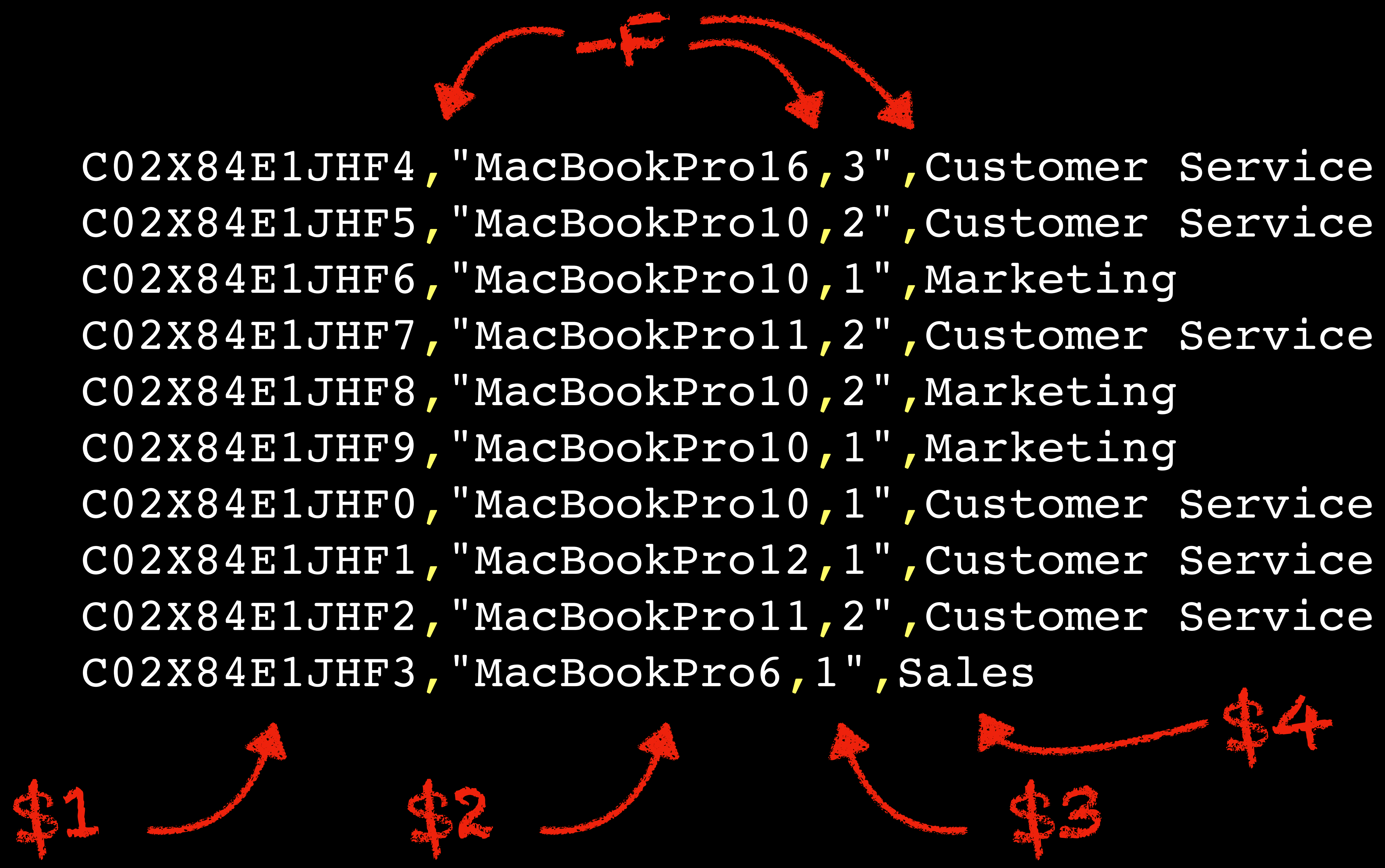
awk

```
awk -F "," '{ print $4 }' assets-list.csv
```

```
C02X84E1JHF4,"MacBookPro16,3",Customer Service
C02X84E1JHF5,"MacBookPro10,2",Customer Service
C02X84E1JHF6,"MacBookPro10,1",Marketing
C02X84E1JHF7,"MacBookPro11,2",Customer Service
C02X84E1JHF8,"MacBookPro10,2",Marketing
C02X84E1JHF9,"MacBookPro10,1",Marketing
C02X84E1JHF0,"MacBookPro10,1",Customer Service
C02X84E1JHF1,"MacBookPro12,1",Customer Service
C02X84E1JHF2,"MacBookPro11,2",Customer Service
C02X84E1JHF3,"MacBookPro6,1",Sales
```

awk

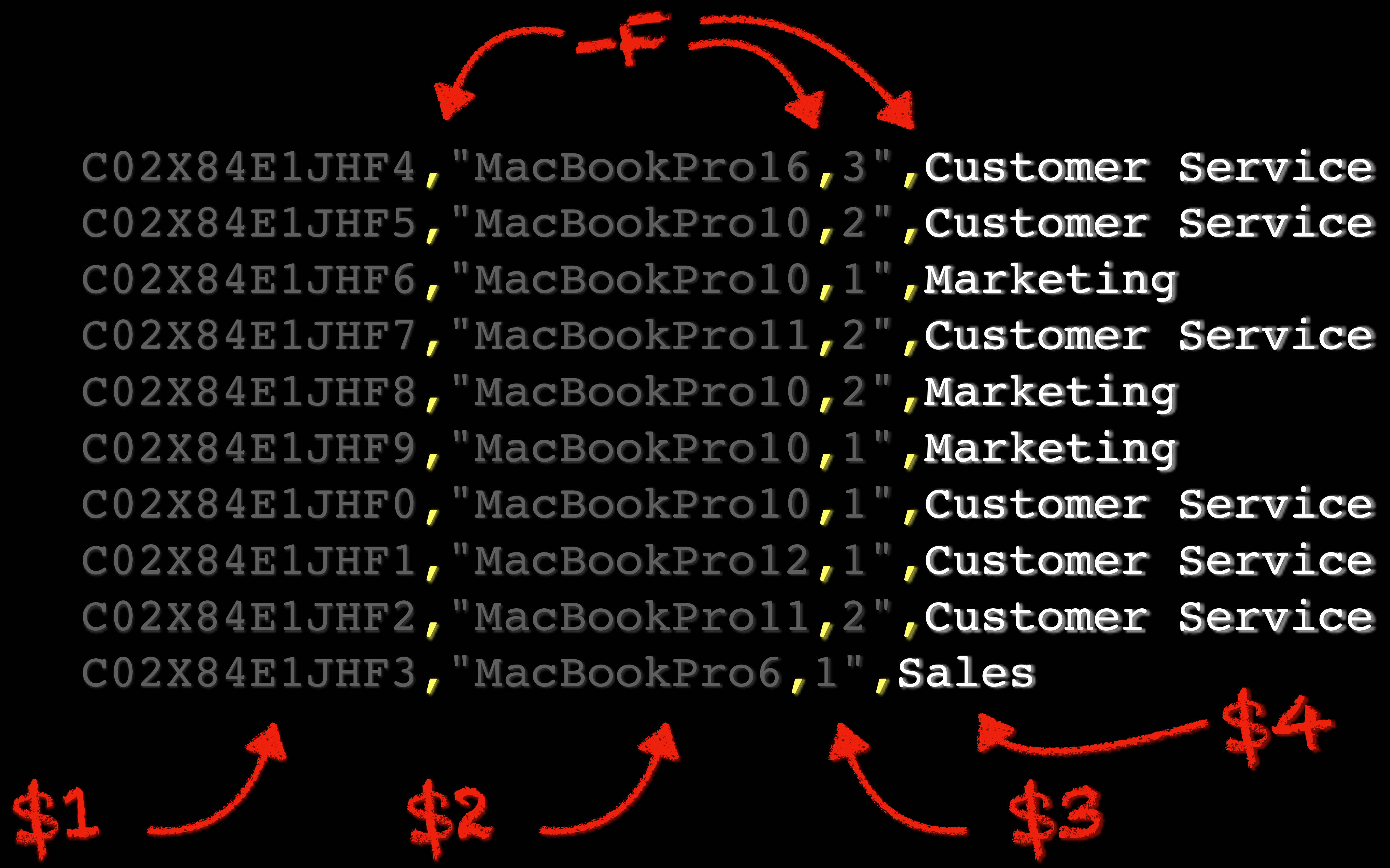
```
awk -F "," '{ print $4 }' assets-list.csv
```



```
C02X84E1JHF4,"MacBookPro16,3",Customer Service
C02X84E1JHF5,"MacBookPro10,2",Customer Service
C02X84E1JHF6,"MacBookPro10,1",Marketing
C02X84E1JHF7,"MacBookPro11,2",Customer Service
C02X84E1JHF8,"MacBookPro10,2",Marketing
C02X84E1JHF9,"MacBookPro10,1",Marketing
C02X84E1JHF0,"MacBookPro10,1",Customer Service
C02X84E1JHF1,"MacBookPro12,1",Customer Service
C02X84E1JHF2,"MacBookPro11,2",Customer Service
C02X84E1JHF3,"MacBookPro6,1",Sales
```

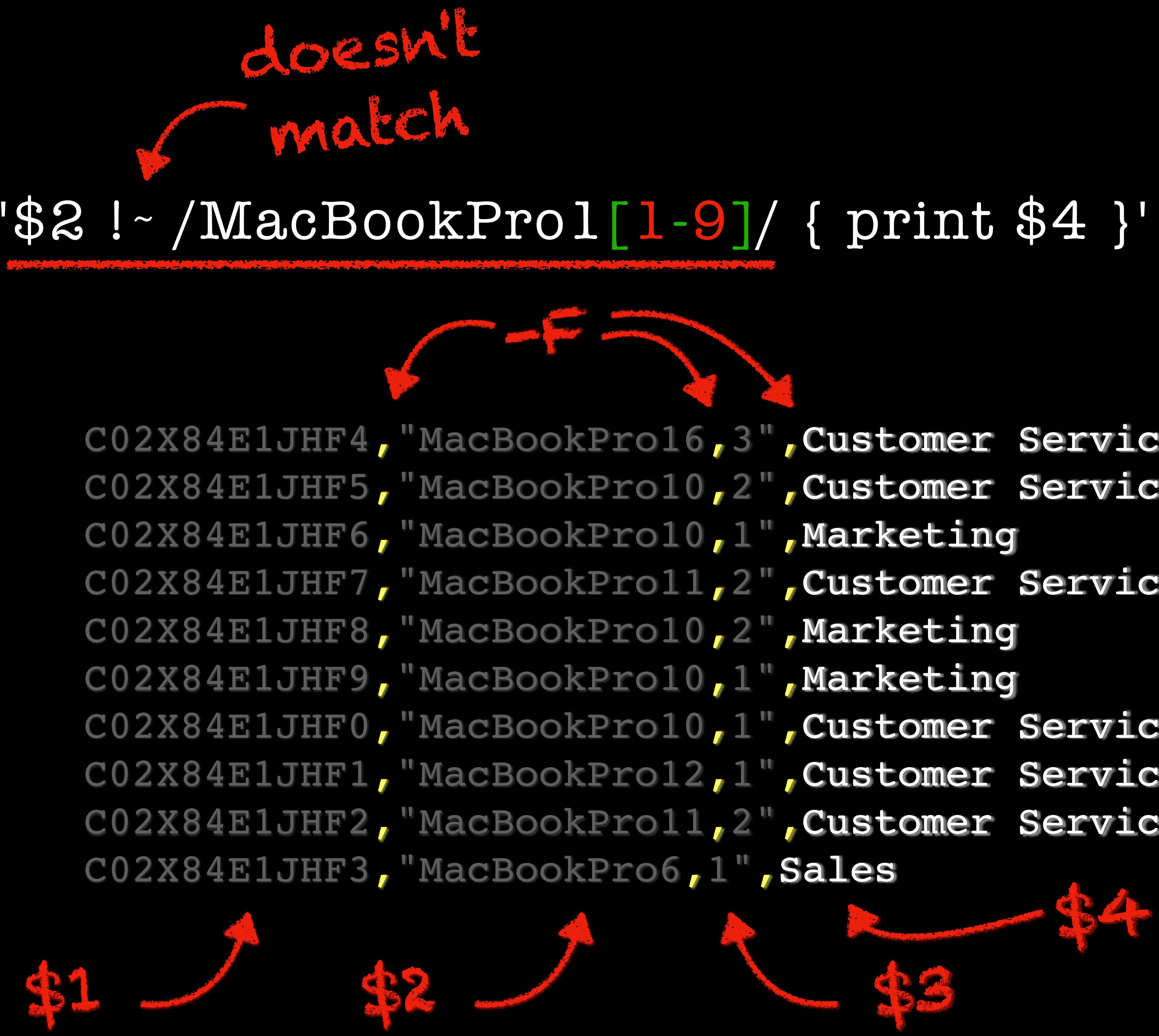
awk

```
awk -F "," '{ print $4 }' assets-list.csv
```



```
C02X84E1JHF4,"MacBookPro16,3",Customer Service
C02X84E1JHF5,"MacBookPro10,2",Customer Service
C02X84E1JHF6,"MacBookPro10,1",Marketing
C02X84E1JHF7,"MacBookPro11,2",Customer Service
C02X84E1JHF8,"MacBookPro10,2",Marketing
C02X84E1JHF9,"MacBookPro10,1",Marketing
C02X84E1JHF0,"MacBookPro10,1",Customer Service
C02X84E1JHF1,"MacBookPro12,1",Customer Service
C02X84E1JHF2,"MacBookPro11,2",Customer Service
C02X84E1JHF3,"MacBookPro6,1",Sales
```

awk



```
awk -F "," '$2 !~ /MacBookPro1[1-9]/ { print $4 }' assets-list.csv
```

```
C02X84E1JHF4,"MacBookPro16,3",Customer Service
C02X84E1JHF5,"MacBookPro10,2",Customer Service
C02X84E1JHF6,"MacBookPro10,1",Marketing
C02X84E1JHF7,"MacBookPro11,2",Customer Service
C02X84E1JHF8,"MacBookPro10,2",Marketing
C02X84E1JHF9,"MacBookPro10,1",Marketing
C02X84E1JHF0,"MacBookPro10,1",Customer Service
C02X84E1JHF1,"MacBookPro12,1",Customer Service
C02X84E1JHF2,"MacBookPro11,2",Customer Service
C02X84E1JHF3,"MacBookPro6,1",Sales
```

awk

doesn't match

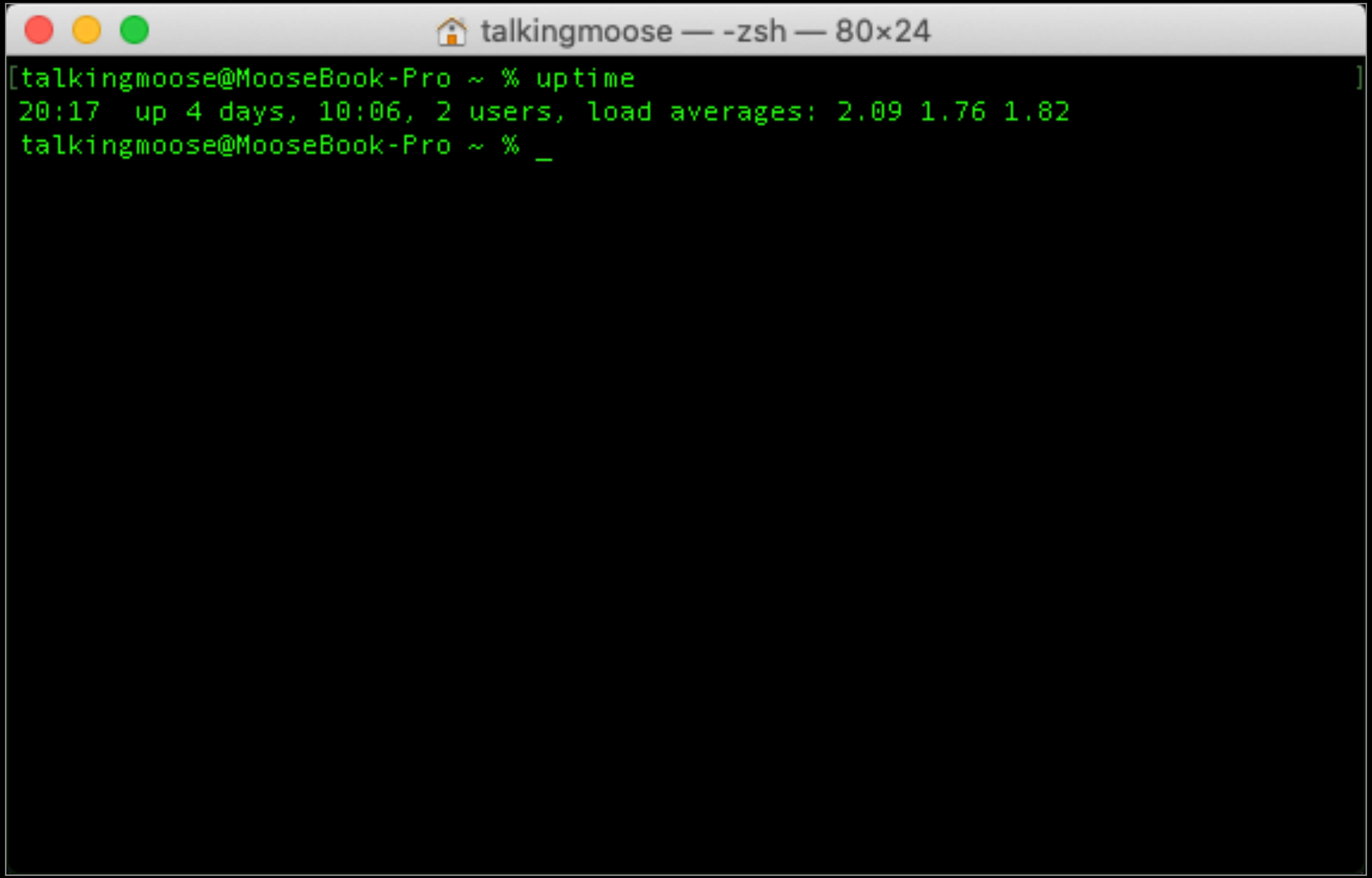
```
awk -F "," '$2 !~ /MacBookPro1[1-9]/ { print $4 }' assets-list.csv
```

-F

```
C02X84E1JHF4,"MacBookPro16,3",Customer Service
C02X84E1JHF5,"MacBookPro10,2",Customer Service
C02X84E1JHF6,"MacBookPro10,1",Marketing
C02X84E1JHF7,"MacBookPro11,2",Customer Service
C02X84E1JHF8,"MacBookPro10,2",Marketing
C02X84E1JHF9,"MacBookPro10,1",Marketing
C02X84E1JHF0,"MacBookPro10,1",Customer Service
C02X84E1JHF1,"MacBookPro12,1",Customer Service
C02X84E1JHF2,"MacBookPro11,2",Customer Service
C02X84E1JHF3,"MacBookPro6,1",Sales
```

$1 $2 $3 $4

awk



```
talkingmoose@MooseBook-Pro ~ % uptime
20:17  up 4 days, 10:06, 2 users, load averages: 2.09 1.76 1.82
talkingmoose@MooseBook-Pro ~ % _
```

awk

```
17:00 up 5 days, 51 mins, 2 users…
17.10 up 5 days, 1:01, 2 users…
17:00 up 51 secs, 2 users…
17:00 up 2 mins, 2 users…
17:00 up 1:01, 2 users…
```

awk

```
uptime | awk -F "(up |, [0-9]+ users)" '{ print $2 }'
```

```
17:00 up 5 days, 51 mins, 2 users…
17.10 up 5 days, 1:01, 2 users…
17:00 up 51 secs, 2 users…
17:00 up 2 mins, 2 users…
17:00 up 1:01, 2 users…
```

sed

sed

```
echo "Martin's MacBook Pro" | sed 's/[^0-9A-Za-z]*//g'
```

```
Martin's MacBook Pro
```

sed

```
echo "Martin's MacBook Pro" | sed 's/[^0-9A-Za-z]*//g'
```

```
MartinsMacBookPro
```

Agenda

What is regex?

Characters with special meanings

Character sets and grouping

Applications and command line tools that support regex

Examples from real world experiences

Regex resources

Agenda

What is regex?

Characters with special meanings

Character sets and grouping

Applications and command line tools that support regex

Examples from real world experiences

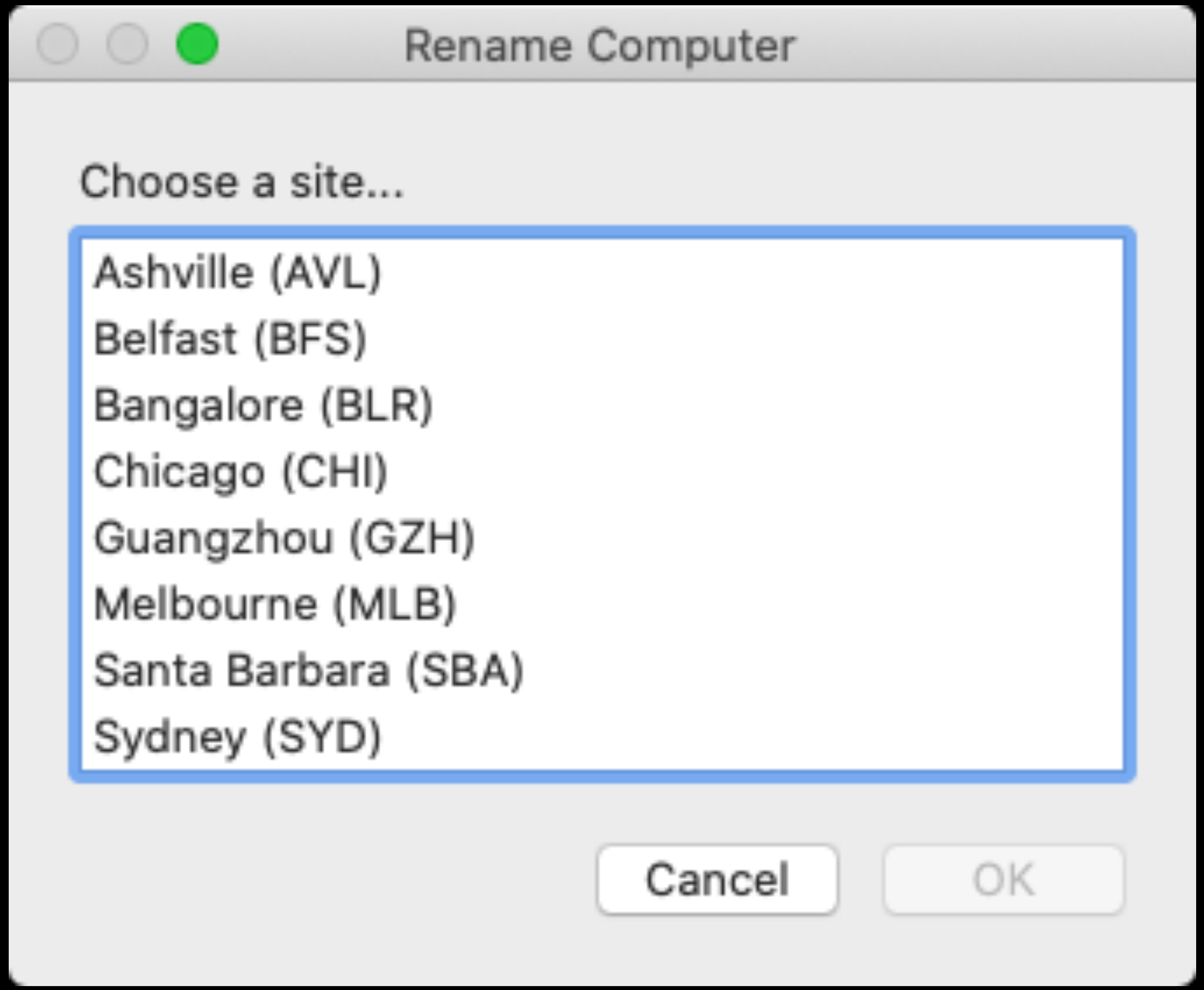
Regex resources

Naming computers

```
CHI-MAC-1JHD3
Prometheus
Admin's MacBook Pro
```

Naming computers

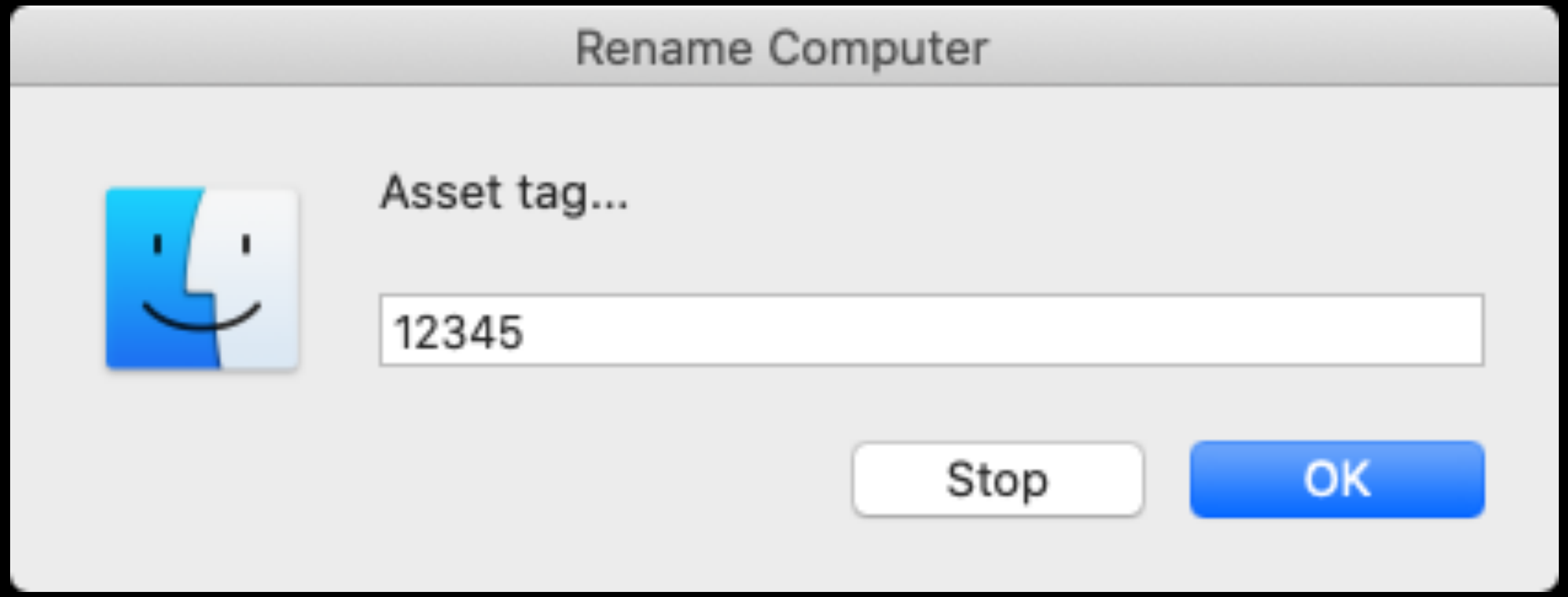
`CHI-MAC-12345`



Rename Computer

Choose a site...

Ashville (AVL)
Belfast (BFS)
Bangalore (BLR)
Chicago (CHI)
Guangzhou (GZH)
Melbourne (MLB)
Santa Barbara (SBA)
Sydney (SYD)



Rename Computer

Asset tag...

12345

Naming computers

Smart Computer Group 1

```
(AVL|BFS|BLR|CHI|GZH|MLB|SBA|SYD)-MAC-[A-Z\d]{5}
```

420

Smart Computer Group 2

```
(AVL|BFS|BLR|CHI|GZH|MLB|SBA|SYD)-MAC-\d{5}
```

20

Smart Computer Group 3

```
Doesn't match smart group 1 and
Doesn't match smart group 2
```

32

452 != 472

Naming computers

Smart Computer Group 1

```
(AVL|BFS|BLR|CHI|GZH|MLB|SBA|SYD)-MAC-\w*[A-Z]\w*
```

400

Smart Computer Group 2

```
(AVL|BFS|BLR|CHI|GZH|MLB|SBA|SYD)-MAC-\d{5}
```

20

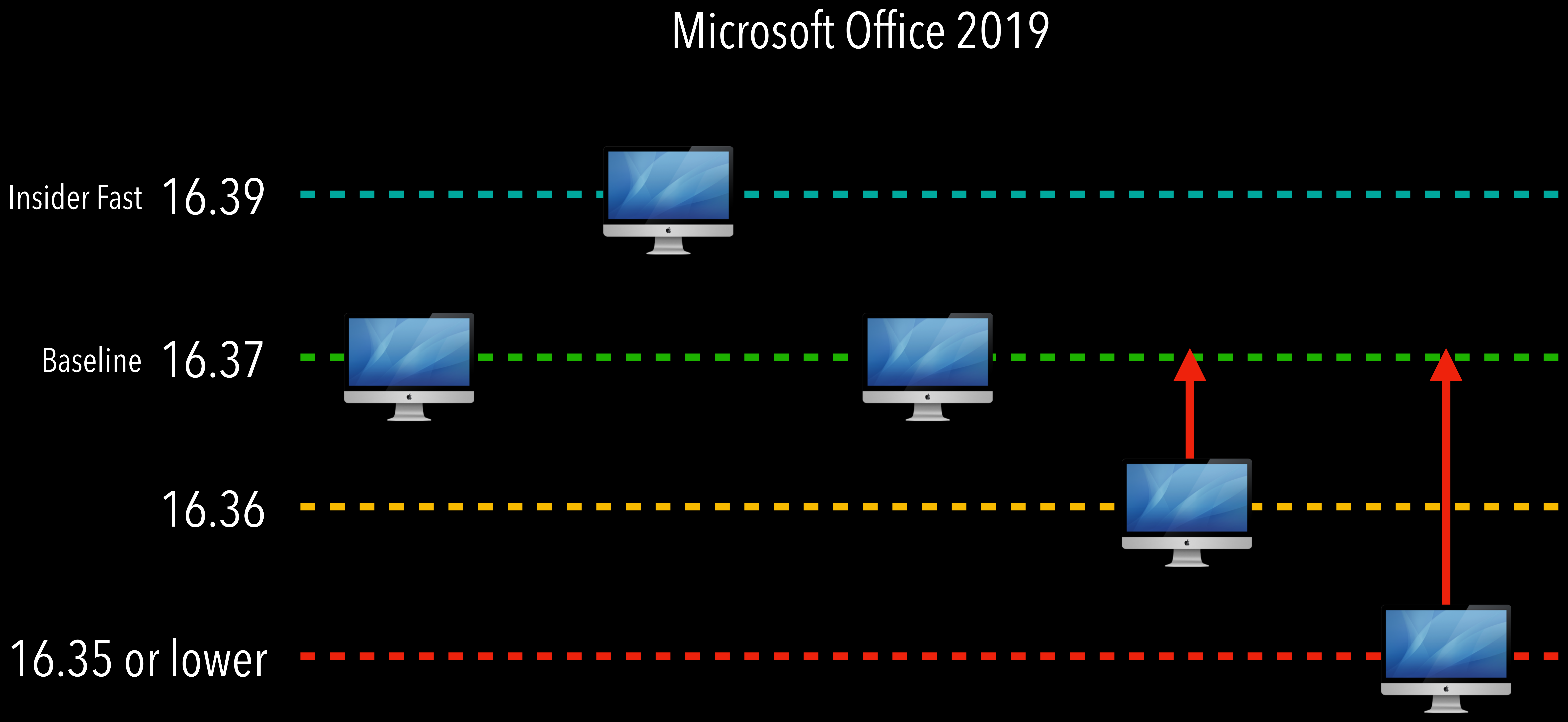
Smart Computer Group 3

```
Doesn't match smart group 1 and
Doesn't match smart group 2
```

32

452 = 452

Software version numbers



Software version numbers

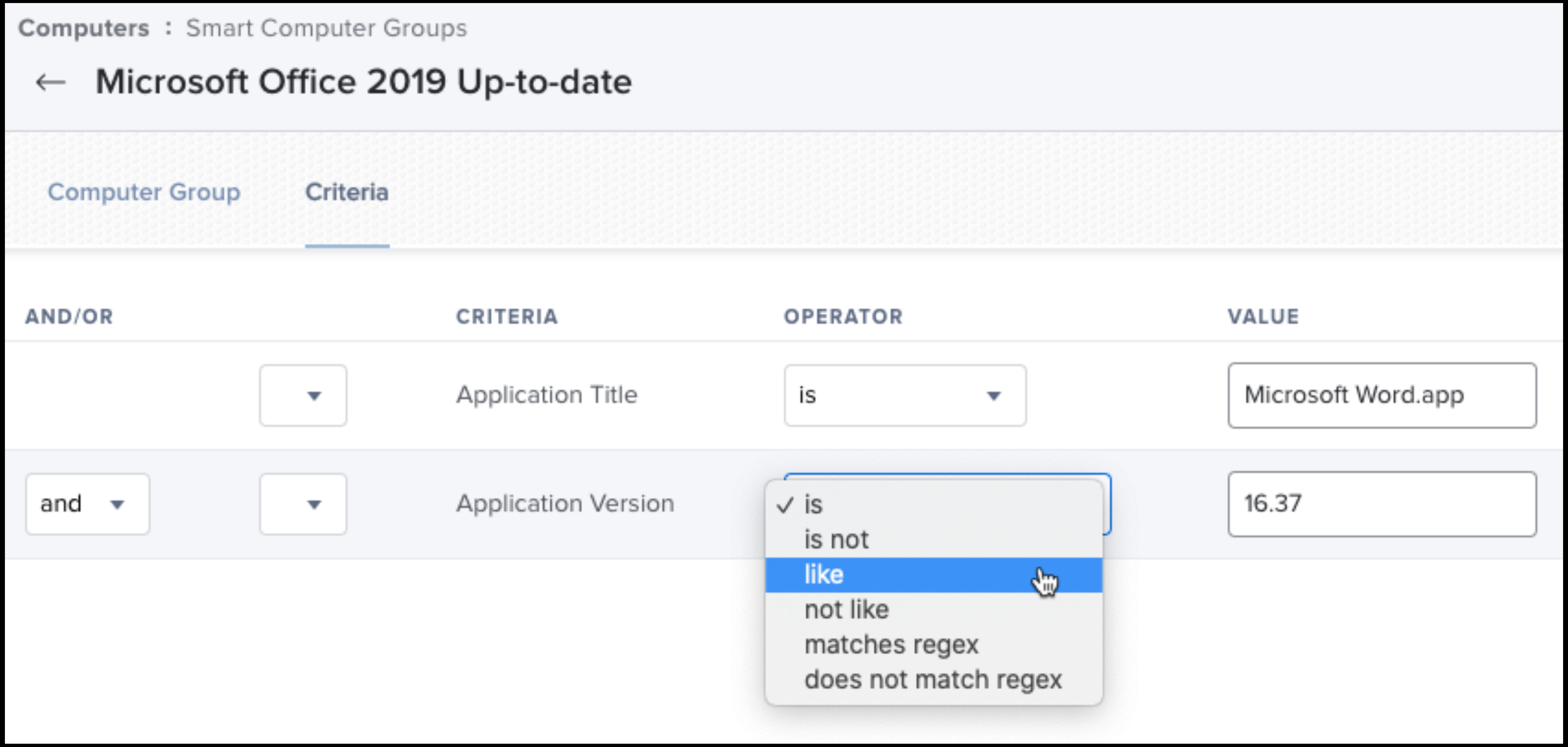
Update all Macs
to 16.37 or higher.

Software version numbers

All Macs $\geq$ 16.37

All Macs $<$ 16.37

Software version ~~numbers~~ strings



Computers : Smart Computer Groups

← Microsoft Office 2019 Up-to-date

Computer Group Criteria

AND/OR		CRITERIA	OPERATOR	VALUE
		Application Title	is	Microsoft Word.app
and		Application Version	is	16.37

Software version ~~numbers~~ strings

16.37 < 16.38

number 16.37 ? 16.37.1 string

Customer: "Of course it's a number!"

Me: "When did any number ever contain two decimals?"

Software version ~~numbers~~ strings

Microsoft Office 2019 – 16.37.1

Google Chrome – 79.0.3945.117

Mozilla Firefox – 74.0.1

Citrix Workspace – 19.10.2.41

Adobe Acrobat Reader DC – 20.006.20034

Adobe Photoshop 2020 – 21.0.3

Microsoft Defender – 100.86.91

Zoom.us – 5.0.3 (24978.0517)

Semantic Versioning – https://semver.org/

Software version ~~numbers~~ strings

Microsoft Office 2019 – 16.37.1

Google Chrome – 79.0.3945.117

Mozilla Firefox – 74.0.1

Citrix Workspace – 19.10.2.41

Adobe Acrobat Reader DC – 20.006.20034

Adobe Photoshop 2020 – 21.0.3

Microsoft Defender – 100.86.91

Zoom.us – 5.0.3 (24978.0517)

Semantic Versioning – https://semver.org/

Software version ~~numbers~~ strings

Google Chrome – 79.0.3945.117

Software version ~~numbers~~ strings

`79.0.3945.117`

Software version ~~numbers~~ strings

```
79.0.3945.11[7-9]
```

Software version ~~numbers~~ strings

```
79.0.3945.1[2-9][7-9]
```

Software version ~~numbers~~ strings

```
79.0.3945.1([2-9]|[7-9])
```

Software version ~~numbers~~ strings

```
79.0.3945.1([2-9]\d|[7-9])
```

Software version ~~numbers~~ strings

```
79.0.3945.1([2-9]\d|1[7-9])
```

Software version ~~numbers~~ strings

```
79.0.3945.117
```

Software version ~~numbers~~ strings

```
79.0.3945.\d{4,}
```

Software version ~~numbers~~ strings

```
79\.0\.3945\.\d{4,}
```

Software version ~~numbers~~ strings

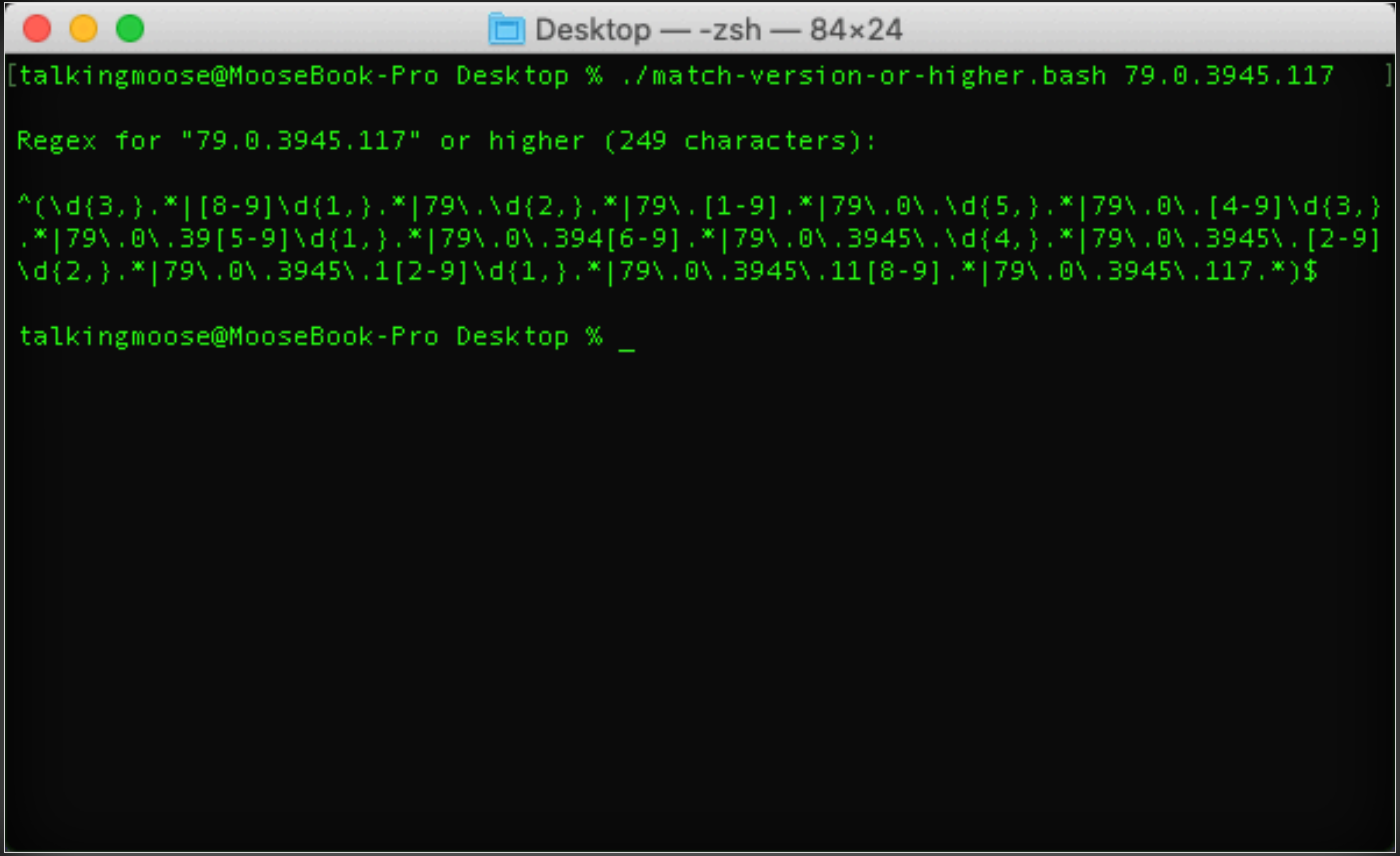
```
79.0.3945.117
```

Software version ~~numbers~~ strings

```
79.0.3945.117
```

```
^(\d{3,}.*|[8-9]\d{1,}.*|79\.\d{2,}.*|79\.
[1-9].*|79\.0\.\d{5,}.*|79\.0\.[4-9]\d{3,}.*|79\.
0\.39[5-9]\d{1,}.*|79\.0\.394[6-9].*|79\.0\.
3945\.\d{4,}.*|79\.0\.3945\.[2-9]\d{2,}.*|79\.
0\.3945\.1[2-9]\d{1,}.*|79\.0\.3945\.11[8-9].*|
79\.0\.3945\.117.*)$
```

Software version ~~numbers~~ strings

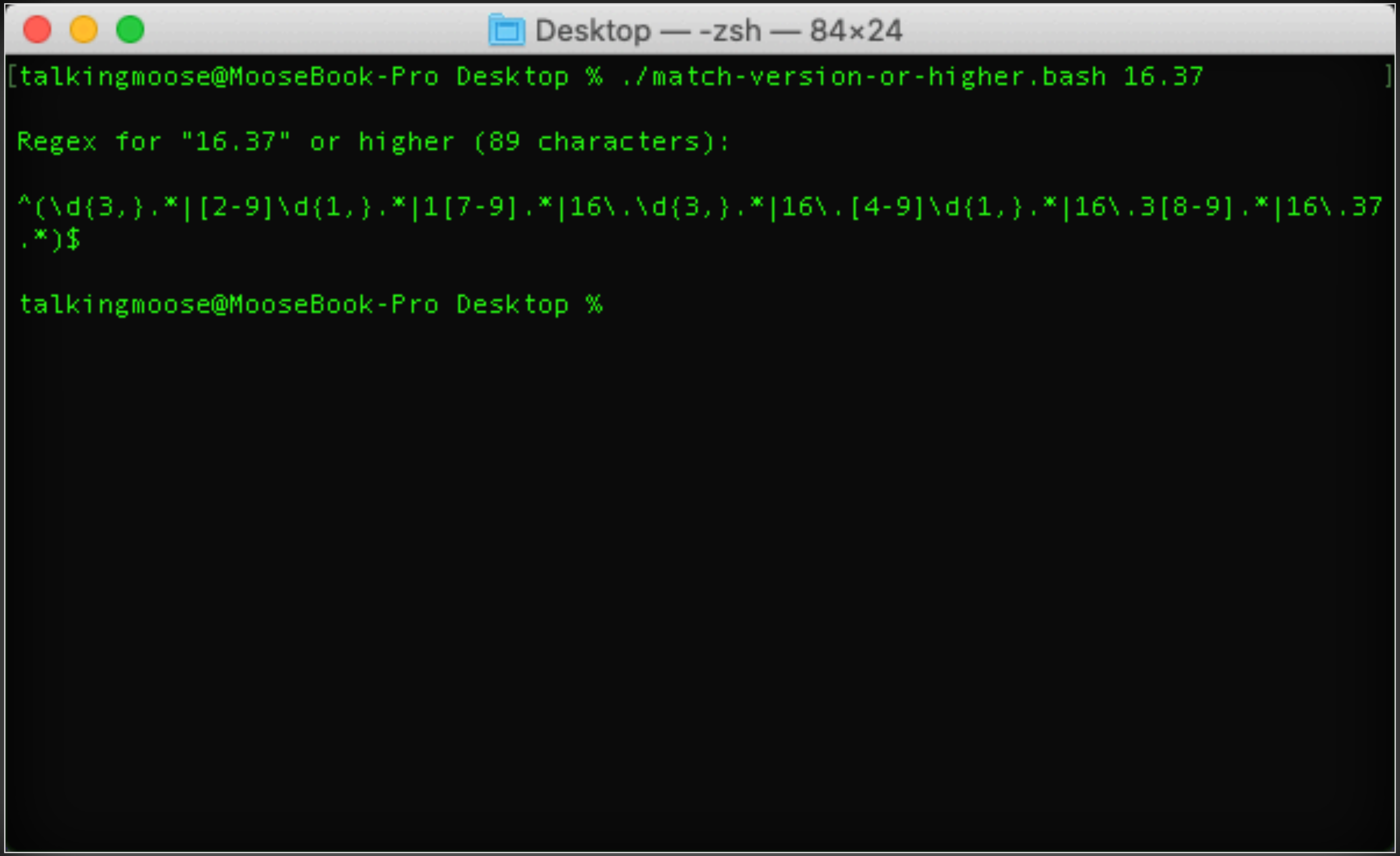


Desktop — -zsh — 84×24

```
[talkingmoose@MooseBook-Pro Desktop % ./match-version-or-higher.bash 79.0.3945.117   ]

Regex for "79.0.3945.117" or higher (249 characters):

^(\d{3,}.*|[8-9]\d{1,}.*|79\.\d{2,}.*|79\.[1-9].*|79\.0\.\d{5,}.*|79\.0\.[4-9]\d{3,}
.*|79\.0\.39[5-9]\d{1,}.*|79\.0\.394[6-9].*|79\.0\.3945\.\d{4,}.*|79\.0\.3945\.[2-9]
\d{2,}.*|79\.0\.3945\.1[2-9]\d{1,}.*|79\.0\.3945\.11[8-9].*|79\.0\.3945\.117.*)$

talkingmoose@MooseBook-Pro Desktop % _
```

Match Version Number or Higher – https://gist.github.com/talkingmoose/2cf20236e665fcd7ec41311d50c89c0e

Software version ~~numbers~~ strings

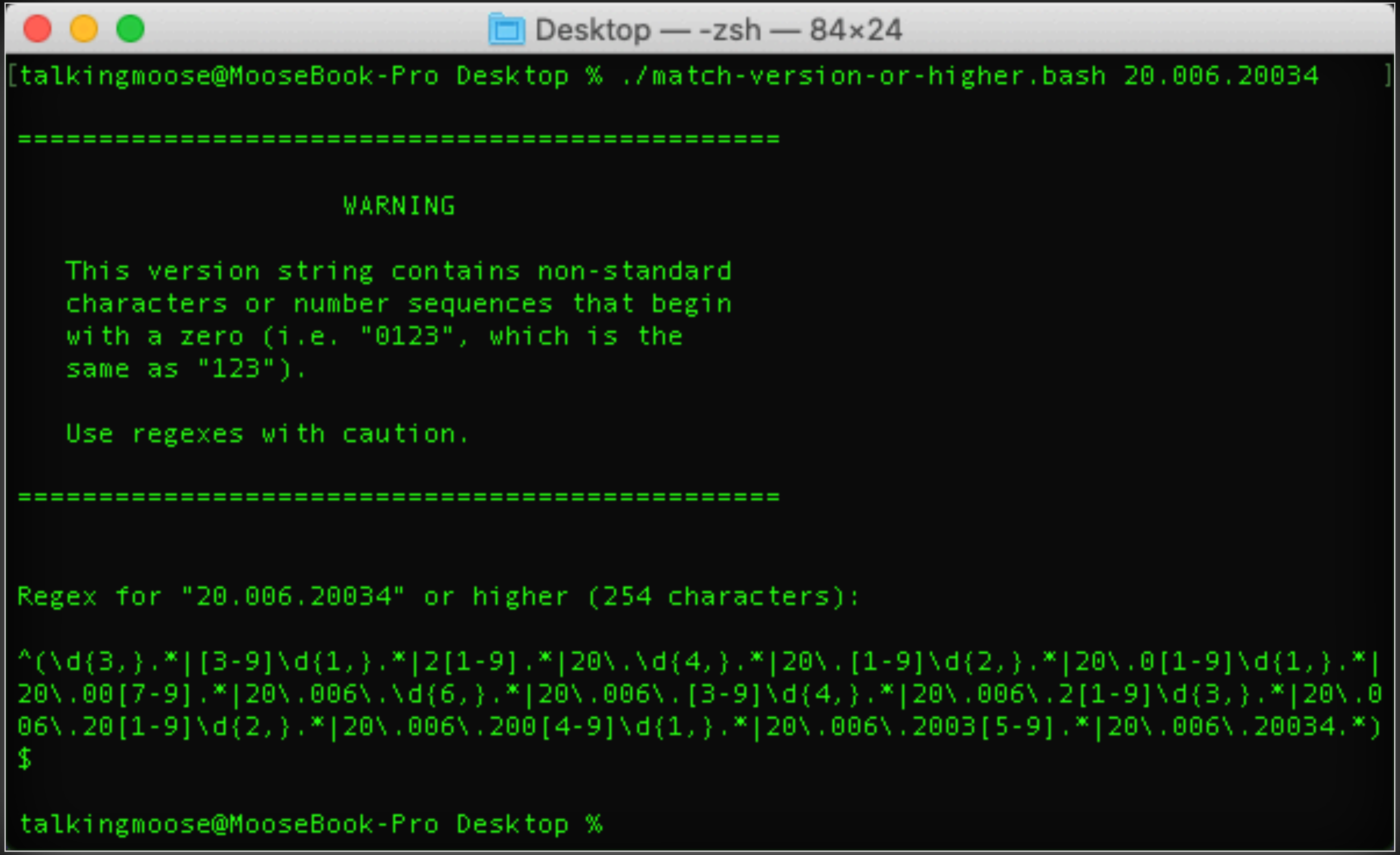


Desktop — -zsh — 84×24

```
[talkingmoose@MooseBook-Pro Desktop % ./match-version-or-higher.bash 16.37          ]

 Regex for "16.37" or higher (89 characters):

 ^(\d{3,}.*|[2-9]\d{1,}.*|1[7-9].*|16\.\d{3,}.*|16\.[4-9]\d{1,}.*|16\.3[8-9].*|16\.37
 .*)$

 talkingmoose@MooseBook-Pro Desktop %
```

Match Version Number or Higher – https://gist.github.com/talkingmoose/2cf20236e665fcd7ec41311d50c89c0e

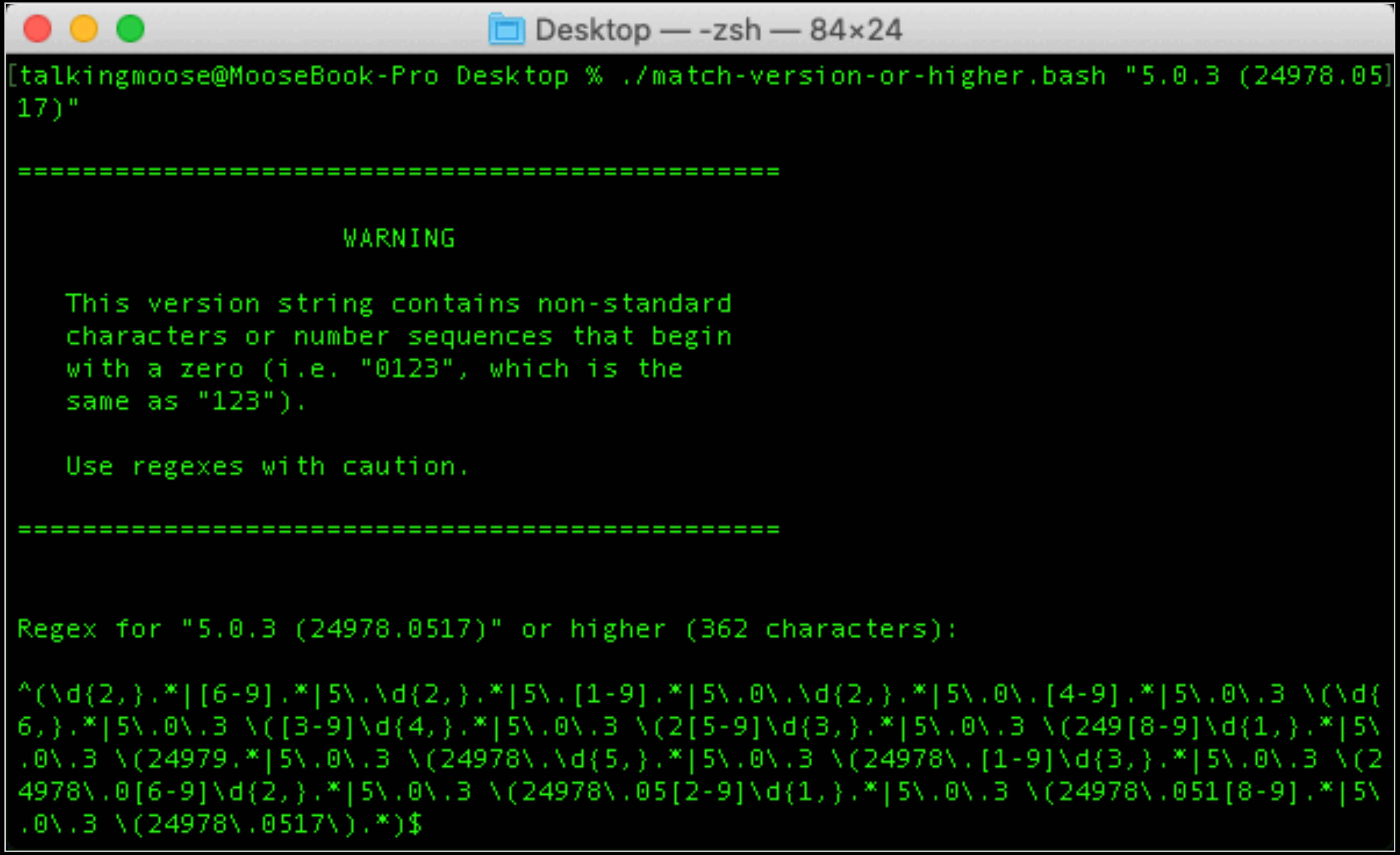
Software version ~~numbers~~ strings



```
[talkingmoose@MooseBook-Pro Desktop % ./match-version-or-higher.bash 20.006.20034      ]

 ===============================================


                    WARNING

    This version string contains non-standard
    characters or number sequences that begin
    with a zero (i.e. "0123", which is the
    same as "123").

    Use regexes with caution.

 ===============================================



 Regex for "20.006.20034" or higher (254 characters):

 ^(\d{3,}.*|[3-9]\d{1,}.*|2[1-9].*|20\.\d{4,}.*|20\.[1-9]\d{2,}.*|20\.0[1-9]\d{1,}.*|
 20\.00[7-9].*|20\.006\.\d{6,}.*|20\.006\.[3-9]\d{4,}.*|20\.006\.2[1-9]\d{3,}.*|20\.0
 06\.20[1-9]\d{2,}.*|20\.006\.200[4-9]\d{1,}.*|20\.006\.2003[5-9].*|20\.006\.20034.*)
 $

 talkingmoose@MooseBook-Pro Desktop %
```

Match Version Number or Higher – https://gist.github.com/talkingmoose/2cf20236e665fcd7ec41311d50c89c0e

Software version ~~numbers~~ strings


Desktop — -zsh — 84×24

```
[talkingmoose@MooseBook-Pro Desktop % ./match-version-or-higher.bash "5.0.3 (24978.05]
17)"

===============================================

                    WARNING

   This version string contains non-standard
   characters or number sequences that begin
   with a zero (i.e. "0123", which is the
   same as "123").

   Use regexes with caution.

===============================================


Regex for "5.0.3 (24978.0517)" or higher (362 characters):

^(\d{2,}.*|[6-9].*|5\.\d{2,}.*|5\.[1-9].*|5\.0\.\d{2,}.*|5\.0\.[4-9].*|5\.0\.3 \(\d{
6,}.*|5\.0\.3 \([3-9]\d{4,}.*|5\.0\.3 \(2[5-9]\d{3,}.*|5\.0\.3 \(249[8-9]\d{1,}.*|5\
.0\.3 \(24979.*|5\.0\.3 \(24978\.\d{5,}.*|5\.0\.3 \(24978\.[1-9]\d{3,}.*|5\.0\.3 \(2
4978\.0[6-9]\d{2,}.*|5\.0\.3 \(24978\.05[2-9]\d{1,}.*|5\.0\.3 \(24978\.051[8-9].*|5\
.0\.3 \(24978\.0517\).*)$
```

Match Version Number or Higher – https://gist.github.com/talkingmoose/2cf20236e665fcd7ec41311d50c89c0e

Agenda

What is regex?

Characters with special meanings

Character sets and grouping

Applications and command line tools that support regex

Examples from real world experiences

Regex resources

Agenda

What is regex?

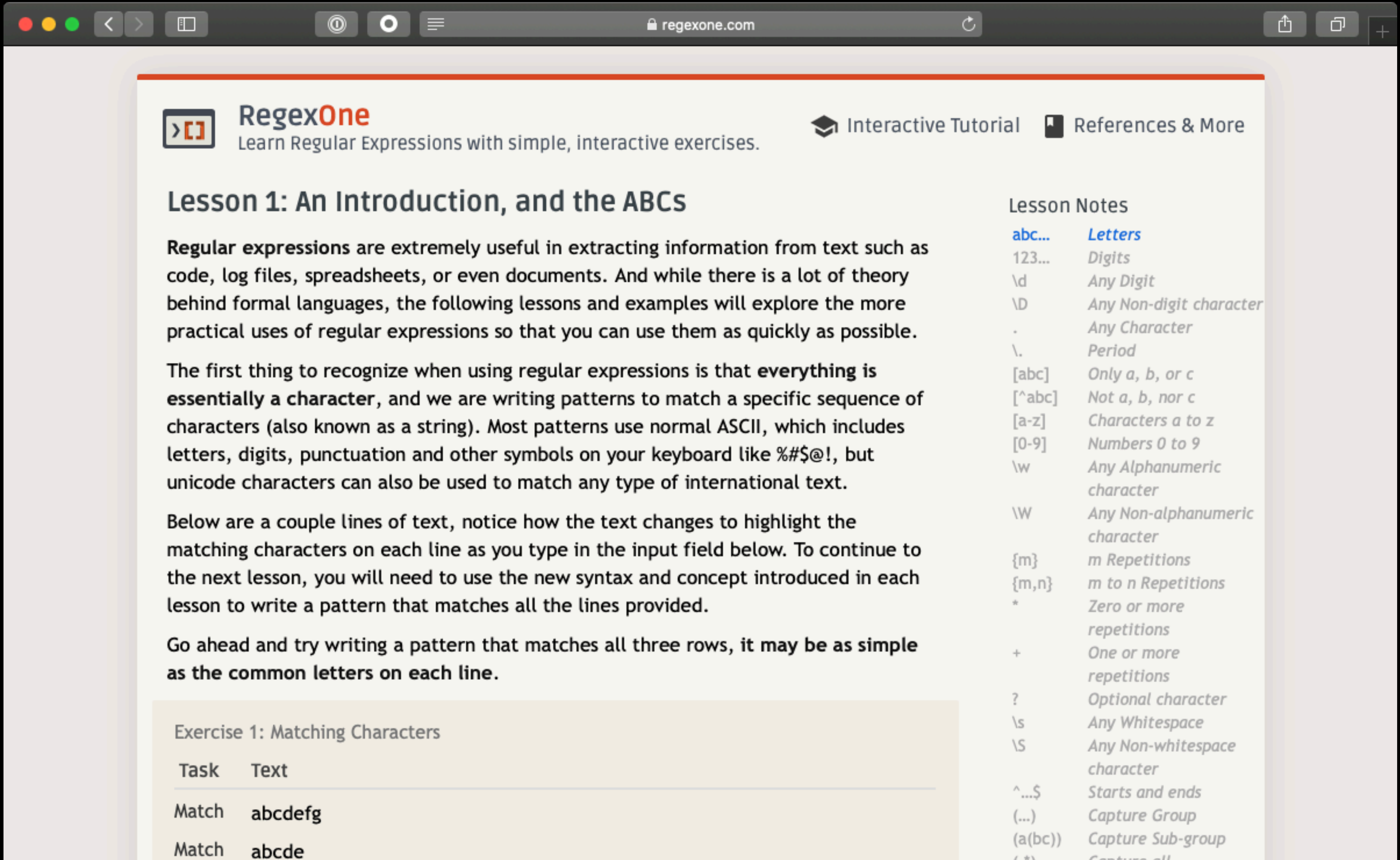
Characters with special meanings

Character sets and grouping

Applications and command line tools that support regex

Examples from real world experiences

Regex resources


regexone.com
RegexOne
Learn Regular Expressions with simple, interactive exercises.
Interactive Tutorial
References & More

Lesson 1: An Introduction, and the ABCs

Regular expressions are extremely useful in extracting information from text such as code, log files, spreadsheets, or even documents. And while there is a lot of theory behind formal languages, the following lessons and examples will explore the more practical uses of regular expressions so that you can use them as quickly as possible.

The first thing to recognize when using regular expressions is that **everything is essentially a character**, and we are writing patterns to match a specific sequence of characters (also known as a string). Most patterns use normal ASCII, which includes letters, digits, punctuation and other symbols on your keyboard like %#$@!, but unicode characters can also be used to match any type of international text.

Below are a couple lines of text, notice how the text changes to highlight the matching characters on each line as you type in the input field below. To continue to the next lesson, you will need to use the new syntax and concept introduced in each lesson to write a pattern that matches all the lines provided.

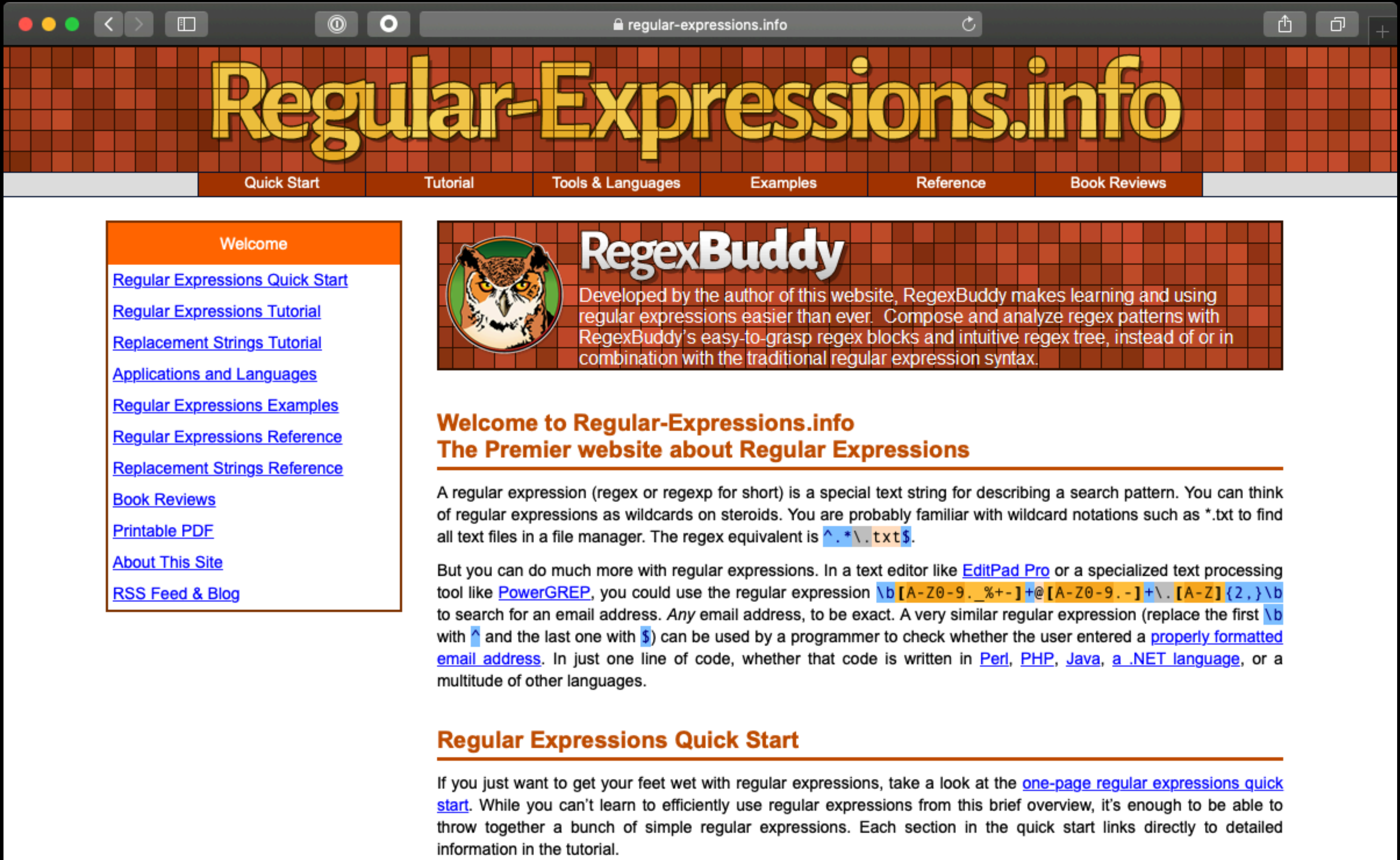
Go ahead and try writing a pattern that matches all three rows, **it may be as simple as the common letters on each line**.

Exercise 1: Matching Characters

Task	Text
Match	abcdefg
Match	abcde

Lesson Notes

abc…	*Letters*
123…	*Digits*
\d	*Any Digit*
\D	*Any Non-digit character*
.	*Any Character*
\.	*Period*
[abc]	*Only a, b, or c*
[^abc]	*Not a, b, nor c*
[a-z]	*Characters a to z*
[0-9]	*Numbers 0 to 9*
\w	*Any Alphanumeric character*
\W	*Any Non-alphanumeric character*
{m}	*m Repetitions*
{m,n}	*m to n Repetitions*
*	*Zero or more repetitions*
+	*One or more repetitions*
?	*Optional character*
\s	*Any Whitespace*
\S	*Any Non-whitespace character*
^…$	*Starts and ends*
(…)	*Capture Group*
(a(bc))	*Capture Sub-group*
]



Welcome

- Regular Expressions Quick Start
- Regular Expressions Tutorial
- Replacement Strings Tutorial
- Applications and Languages
- Regular Expressions Examples
- Regular Expressions Reference
- Replacement Strings Reference
- Book Reviews
- Printable PDF
- About This Site
- RSS Feed & Blog

Developed by the author of this website, RegexBuddy makes learning and using regular expressions easier than ever. Compose and analyze regex patterns with RegexBuddy's easy-to-grasp regex blocks and intuitive regex tree, instead of or in combination with the traditional regular expression syntax.

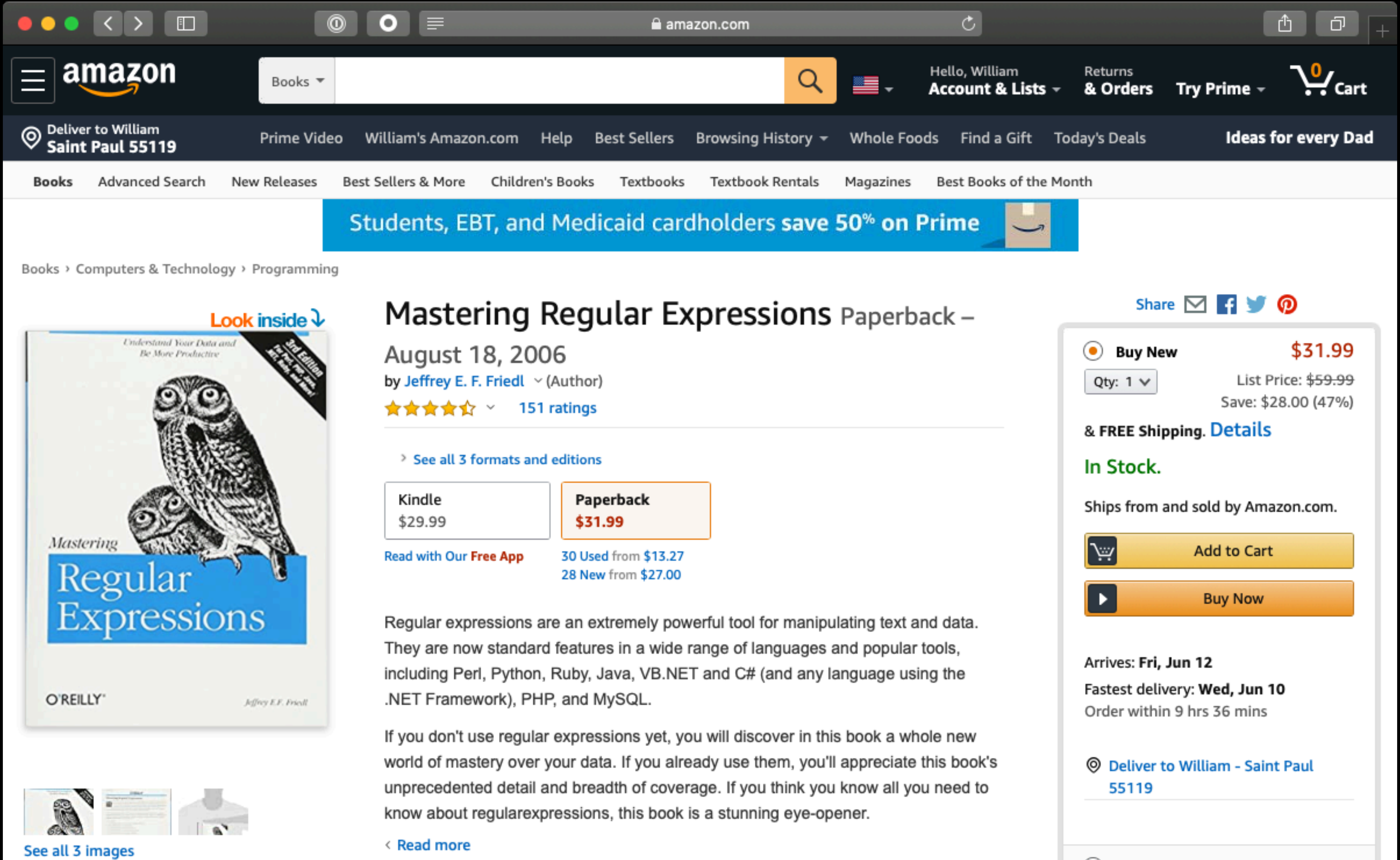
Welcome to Regular-Expressions.info
The Premier website about Regular Expressions

A regular expression (regex or regexp for short) is a special text string for describing a search pattern. You can think of regular expressions as wildcards on steroids. You are probably familiar with wildcard notations such as *.txt to find all text files in a file manager. The regex equivalent is `^.*\.txt$`.

But you can do much more with regular expressions. In a text editor like EditPad Pro or a specialized text processing tool like PowerGREP, you could use the regular expression `\b[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,}\b` to search for an email address. *Any* email address, to be exact. A very similar regular expression (replace the first `\b` with `^` and the last one with `$`) can be used by a programmer to check whether the user entered a properly formatted email address. In just one line of code, whether that code is written in Perl, PHP, Java, a .NET language, or a multitude of other languages.

Regular Expressions Quick Start

If you just want to get your feet wet with regular expressions, take a look at the one-page regular expressions quick start. While you can't learn to efficiently use regular expressions from this brief overview, it's enough to be able to throw together a bunch of simple regular expressions. Each section in the quick start links directly to detailed information in the tutorial.



amazon.com

Books

Hello, William
Account & Lists

Returns
& Orders

Try Prime

0 Cart

Deliver to William
Saint Paul 55119

Prime Video | William's Amazon.com | Help | Best Sellers | Browsing History | Whole Foods | Find a Gift | Today's Deals | Ideas for every Dad

Books | Advanced Search | New Releases | Best Sellers & More | Children's Books | Textbooks | Textbook Rentals | Magazines | Best Books of the Month

Students, EBT, and Medicaid cardholders save 50% on Prime

Books › Computers & Technology › Programming

Look inside

Mastering Regular Expressions Paperback – August 18, 2006

by Jeffrey E. F. Friedl (Author)

151 ratings

See all 3 formats and editions

Kindle	Paperback
$29.99	$31.99
Read with Our Free App	30 Used from $13.27
	28 New from $27.00

Regular expressions are an extremely powerful tool for manipulating text and data. They are now standard features in a wide range of languages and popular tools, including Perl, Python, Ruby, Java, VB.NET and C# (and any language using the .NET Framework), PHP, and MySQL.

If you don't use regular expressions yet, you will discover in this book a whole new world of mastery over your data. If you already use them, you'll appreciate this book's unprecedented detail and breadth of coverage. If you think you know all you need to know about regularexpressions, this book is a stunning eye-opener.

Read more

See all 3 images

Share

Buy New $31.99

Qty: 1

List Price: ~~$59.99~~
Save: $28.00 (47%)

& FREE Shipping. Details

In Stock.

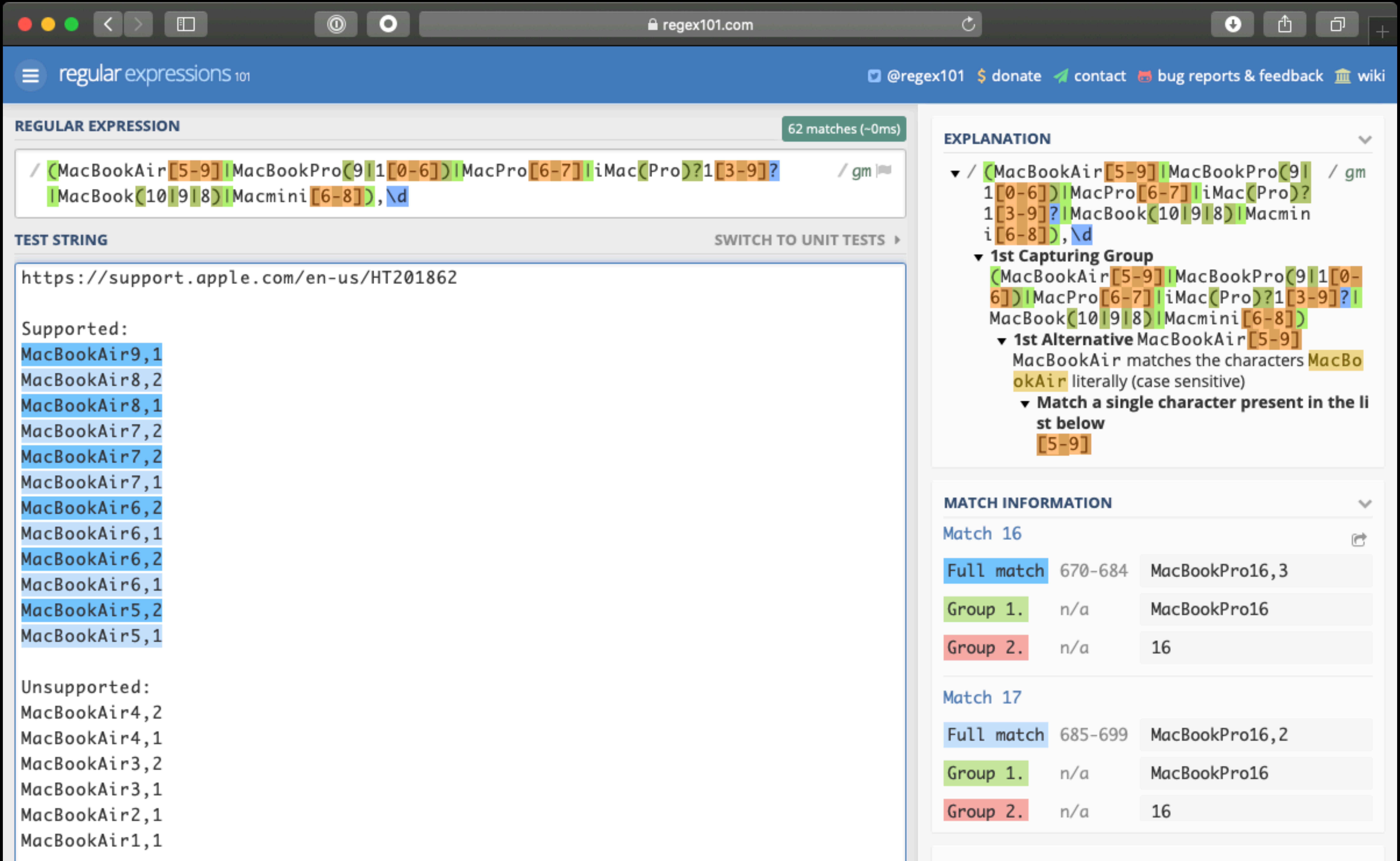
Ships from and sold by Amazon.com.

Add to Cart

Buy Now

Arrives: Fri, Jun 12
Fastest delivery: Wed, Jun 10
Order within 9 hrs 36 mins

Deliver to William - Saint Paul 55119



regex101.com

regular expressions 101

@regex101 $ donate contact bug reports & feedback wiki

REGULAR EXPRESSION

62 matches (~0ms)

```
/ (MacBookAir[5-9]|MacBookPro(9|1[0-6])|MacPro[6-7]|iMac(Pro)?1[3-9]?|MacBook(10|9|8)|Macmini[6-8]),\d / gm
```

TEST STRING

SWITCH TO UNIT TESTS ›

```
https://support.apple.com/en-us/HT201862

Supported:
MacBookAir9,1
MacBookAir8,2
MacBookAir8,1
MacBookAir7,2
MacBookAir7,2
MacBookAir7,1
MacBookAir6,2
MacBookAir6,1
MacBookAir6,2
MacBookAir6,1
MacBookAir5,2
MacBookAir5,1

Unsupported:
MacBookAir4,2
MacBookAir4,1
MacBookAir3,2
MacBookAir3,1
MacBookAir2,1
MacBookAir1,1
```

EXPLANATION

```
/ (MacBookAir[5-9]|MacBookPro(9|1[0-6])|MacPro[6-7]|iMac(Pro)?1[3-9]?|MacBook(10|9|8)|Macmini[6-8]),\d / gm
```

1st Capturing Group (MacBookAir[5-9]|MacBookPro(9|1[0-6])|MacPro[6-7]|iMac(Pro)?1[3-9]?|MacBook(10|9|8)|Macmini[6-8])

1st Alternative MacBookAir[5-9]

MacBookAir matches the characters MacBookAir literally (case sensitive)

Match a single character present in the list below

[5-9]

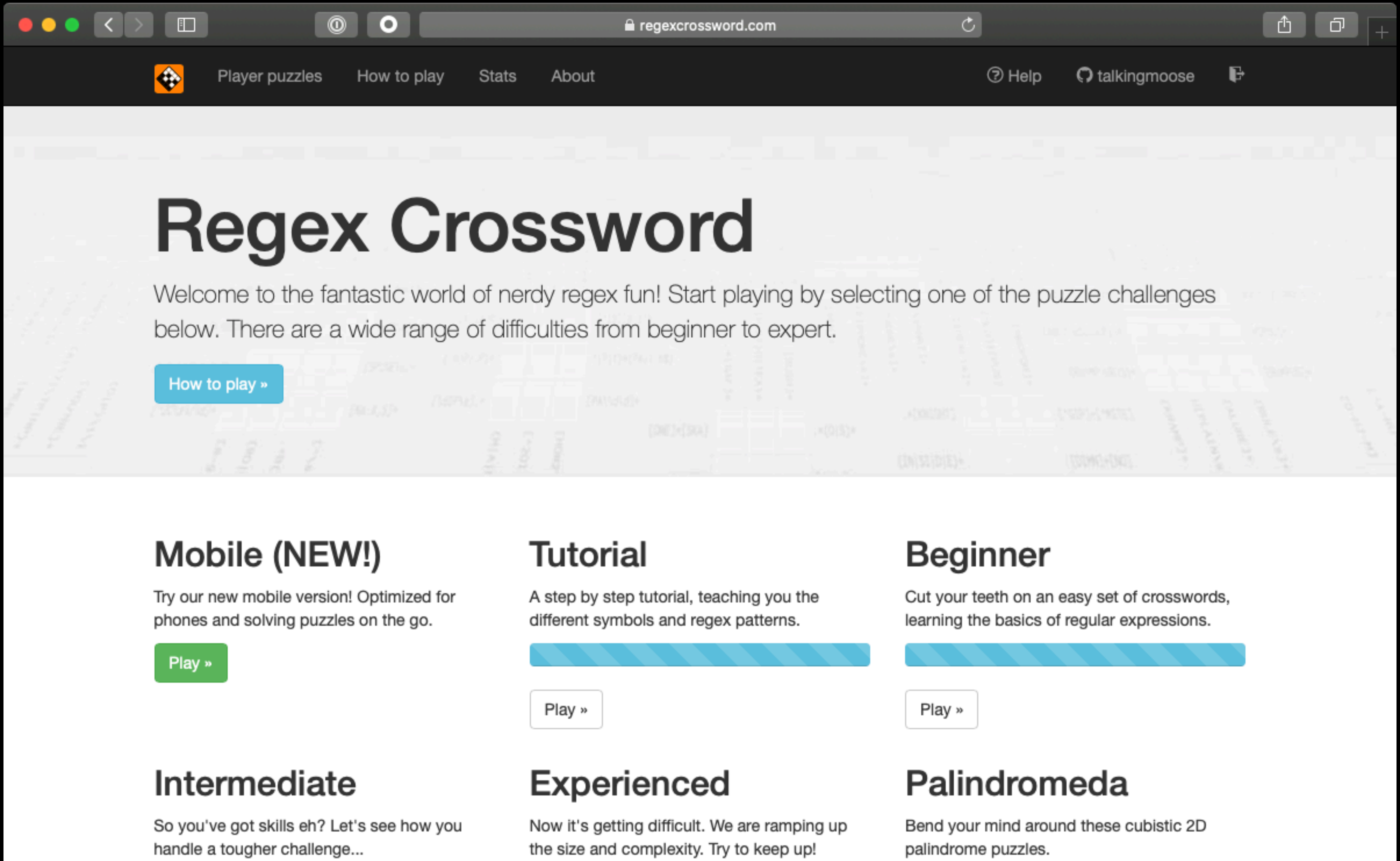
MATCH INFORMATION

Match 16

Full match	670-684	MacBookPro16,3
Group 1.	n/a	MacBookPro16
Group 2.	n/a	16

Match 17

Full match	685-699	MacBookPro16,2
Group 1.	n/a	MacBookPro16
Group 2.	n/a	16



regexcrossword.com

Player puzzles How to play Stats About

Help talkingmoose

Regex Crossword

Welcome to the fantastic world of nerdy regex fun! Start playing by selecting one of the puzzle challenges below. There are a wide range of difficulties from beginner to expert.

How to play »

Mobile (NEW!)

Try our new mobile version! Optimized for phones and solving puzzles on the go.

Play »

Tutorial

A step by step tutorial, teaching you the different symbols and regex patterns.

Play »

Beginner

Cut your teeth on an easy set of crosswords, learning the basics of regular expressions.

Play »

Intermediate

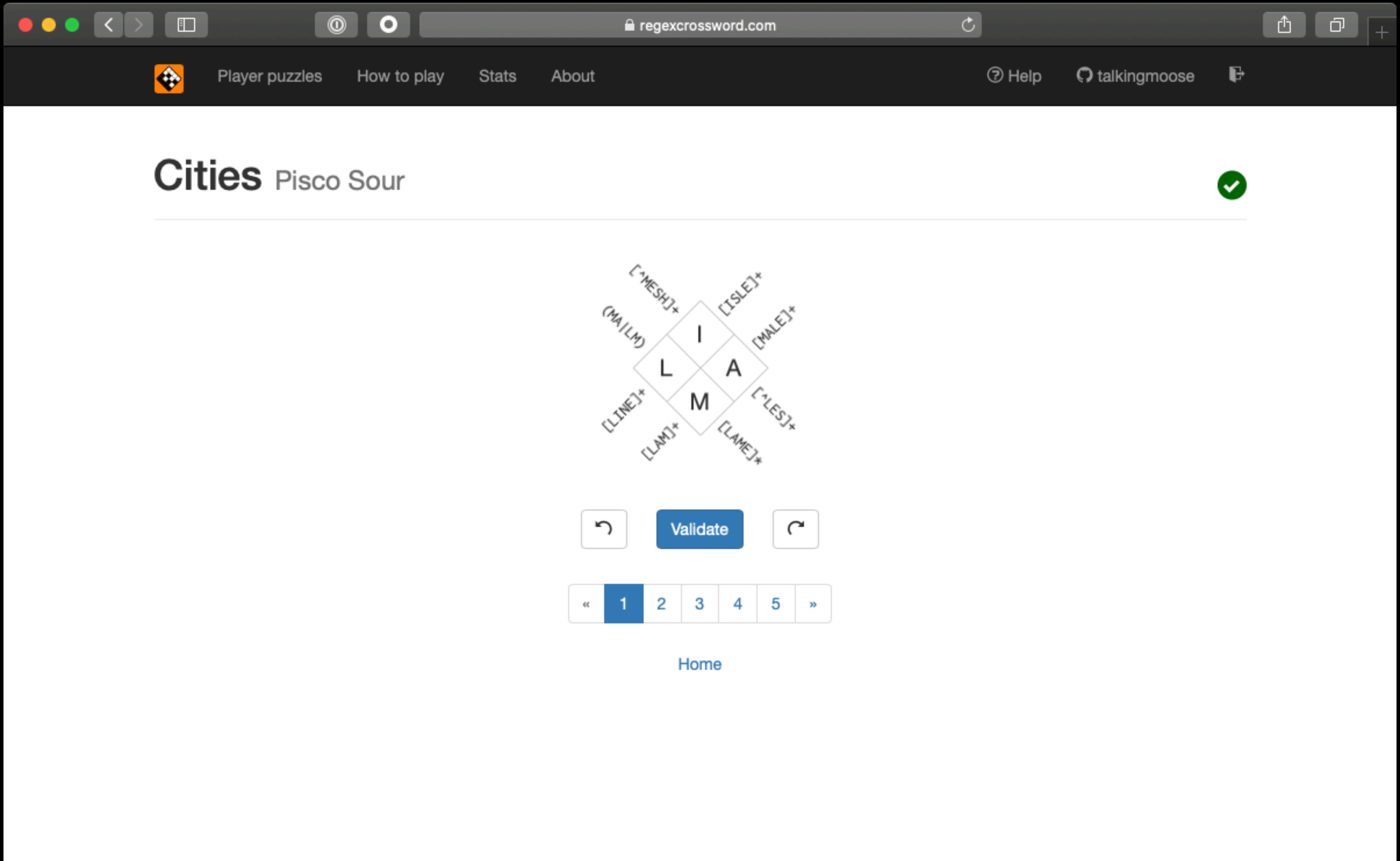
So you've got skills eh? Let's see how you handle a tougher challenge...

Experienced

Now it's getting difficult. We are ramping up the size and complexity. Try to keep up!

Palindromeda

Bend your mind around these cubistic 2D palindrome puzzles.


regexcrossword.com

Player puzzles How to play Stats About

Help talkingmoose

Cities Pisco Sour

Validate

« 1 2 3 4 5 »

Home

Agenda

What is regex?

Characters with special meanings

Character sets and grouping

Applications and command line tools that support regex

Examples from real world experiences

Regex resources

An Introduction to

`(re.ex|re+gex|re?gex|re*gex){1}`

William Smith

Professional Services Enginerd, Jamf

@talkingmoose *the Slacks*

@meck *the Twitters*

bill@talkingmoose.net *the inboxen*

Resources:

https://github.com/talkingmoose/introduction-to-regex

"Regular Expressions"

WHENEVER I LEARN A NEW SKILL I CONCOCT ELABORATE FANTASY SCENARIOS WHERE IT LETS ME SAVE THE DAY.
OH NO! THE KILLER MUST HAVE FOLLOWED HER ON VACATION!
BUT TO FIND THEM WE'D HAVE TO SEARCH THROUGH 200 MB OF EMAILS LOOKING FOR SOMETHING FORMATTED LIKE AN ADDRESS!
IT'S HOPELESS!
EVERYBODY STAND BACK.
I KNOW REGULAR EXPRESSIONS.
PERL!
TAP TAP

https://xkcd.com/208/